RUPRECHT-KARLS-
UNIVERSITÄT
HEIDELBERG

HOCHSCHULE HEILBRONN
TECHNIK WIRTSCHAFT INFORMATIK

Universität Heidelberg - Hochschule Heilbronn

Evaluation und Anwendung aktueller Entwicklungen im Bereich Bluetooth Low Energy am Beispiel von iBeacon

Masterarbeit
Medizinische Informatik
-
Andreas Kudak

Referent: Prof. Dr. Martin Haag, Hochschule Heilbronn
Korreferent: Prof. Dr.-Ing. Gerrit Meixner, Hochschule Heilbronn

Danksagung

Ich möchte mich bei all denjenigen bedanken, die mich bei der Erstellung dieser Masterarbeit unterstützt haben.
Mein besonderer Dank gilt Prof. Dr. Martin Haag, der mich durch seine hilfreichen Anregungen unterstützt hat und Dipl.-Inform. Med. Jörn Heid, der mir mit seinem technischem Know-How wertvolle Hinweise gegeben hat.

Inhaltsverzeichnis

1 Einleitung

1.1 Gegenstand und Motivation

Die bereits in den 1970er-Jahren [1] entwickelte Technologie NAVSTAR GPS[1] (GPS) arbeitet elektronisch und eignet sich besonders zur Navigation in großen und freiliegenden Gebieten, da sie die Standorte der Endgeräte satellitengestützt errechnet. In Gebäuden stößt diese Technik aufgrund der Signaldämpfung durch Hindernisse, wie Decken, schnell an ihre Grenzen. GPS-Signale sind im Inneren eines Gebäudes um 20-30 dB (Faktor 100-1000) schwächer, als im Außenraum [2]. Auch die Differenzierung verschiedener Etagen ist mittels der GPS-Technologie nicht möglich.

Die herkömmliche Navigation in Gebäuden wird zurzeit häufig mittels Pfeilen und Karten realisiert. Aufgrund immer größer werdender Gebäude und dem Wunsch, ortsbezogene Informationen zu übermitteln, steigt die Nachfrage einer elektronischen Indoor-Navigation stetig.

Eine mögliche Lösung zur Indoor-Navigation bietet iBeacon. Dieser proprietäre Standard wurde 2013 von der Apple Inc.[2] eingeführt und basiert auf der Bluetooth Low Energy-Technologie (BLE), welche durch einen geringen Energieverbrauch gekennzeichnet ist. Ursprünglich dient iBeacon dazu, einem BLE fähigem Endgerät mitzuteilen, dass es sich in der Reichweite eines iBeacons befindet. Das Endgerät kann so ortsspezifische Informationen an den Benutzer übermitteln, da das Endgerät den Standort des iBeacons kennt. So können z.B. Informationen zu einem Patienten eines Krankenhauses an den Tabletcomputer (Tablet) des medizinischen Personals übermittelt werden. Dies geschieht durch eine Abstandsmessung zwischen dem iBeacon und dem Endgerät. Unter der Hinzunahme zwei weiterer iBeacons kann theoretisch eine Positionsbestimmung des Mobiltelefons und damit eine Navigation durchgeführt werden.

---

[1] Navigational Satellite Timing and Ranging – Global Positioning System

[2] www.apple.com

1.2 Problemstellung und Zielsetzung

Aufgrund der Tatsache, dass iBeacon erst 2013 von Apple auf der Worldwide Developers Conference (WWDC) vorgestellt wurde, gibt es zurzeit wenige Anwendungen und kaum Fachliteratur.
Der Mangel an Anwendungen geht einher mit einer geringen Anzahl von Messungen und Erfahrungen zu der Genauigkeit der Abstandsbestimmung zweier Geräte. Des Weiteren ist für das durch Apple gedachte Einsatzgebiet von iBeacon bei größerer Entfernung keine genaue Abstandsmessung nötig. Vielmehr geht es darum, ob sich der Benutzer in der Nähe eines iBeacon befindet. So wird die Entfernung durch Apple in unknown, immediate, near und far klassifiziert [3]. Eine niedrige Genauigkeit könnte die Positionsbestimmung erschweren und dadurch das Navigationssystem ungenau und damit unbrauchbar machen.
Um zu klären, ob iBeacon zur effektiven Indoor-Navigation in Frage kommt, sollen zuerst die Technologien Bluetooth, Bluetooth Low Energy und iBeacon gegeneinander abgegrenzt, sowie ein jeweiliger Mehrwert evaluiert werden.
Anschließend soll mit möglichst vielen, unterschiedlichen Geräten und Software, die Genauigkeit in der Distanzmessung bestimmt werden. Auch unter der Hinzunahme von Hindernissen, wie Mauern oder Personen. Am Ende soll ein auf iBeacon basierender Prototyp zur Indoor-Navigation stehen.

2 Theoretische Grundlagen

2.1 Bluetooth

Der Industriestandard Bluetooth wurde in den 1990er-Jahren durch die Bluetooth Special Interest Group Inc. (SIG) eingeführt [4]. Seitdem dient die Technologie der Datenübertragung zwischen verschiedenen Endgeräten via Funktechnik. Je nach Sendeleistung ist dabei eine Reichweite von 1-100 Metern [5, 6] möglich. Der Name Bluetooth entstand aus der Idee, unterschiedliche Geräte miteinander zu verbinden. Der Name leitet sich vom dänischen König Harald Blåtand (im englischen Bluetooth) ab, welcher verfeindete Teile von Norwegen und Dänemark vereinte [7].
Das klassische Bluetooth erreicht in der Version 1.0 zur Übertragung eine Basic Rate (BR) von 1 MBit/s. Die Version 2.0 erreicht bereits eine Enhanced Data Rate (EDR) von 2-3 MBit/s. [8] In der Version 3.0 aus dem Jahre 2009 ist in Kombination mit der WLAN-Übertragungstechnik IEEE 802.11g eine Übertragungsgeschwindigkeit von 24 Mbps möglich [9]. Dies bietet eine große Vielschichtigkeit von Anwendungsmöglichkeiten. So ist Bluetooth im Jahr 2013 in 2,5 Milliarden [4] Produkten, wie Mobiltelefonen, Computern, medizinischen Geräten oder Unterhaltungsmedien verfügbar [6]. Aufgrund der weiten Verbreitung und der Tatsache, dass die SIG im Jahre 2013 etwa 20.000 Unternehmen als Mitglieder aufweist [4], lässt sich sagen, dass sich Bluetooth als Standard zum Datenaustausch verschiedener Geräte etabliert hat.

2.2 Bluetooth Low Energy

Bluetooth Low Energy wurde im Juli 2010 als Teil von Bluetooth 4.0 in die Bluetooth Technologie integriert. Frühere Namen für Bluetooth Low Energy waren Wibree und Ultra Low Power (ULP), diese wurden jedoch von der SIG verworfen. Heute sind auch die Abkürzungen BLE und LE geläufig. Wie der Name schon andeutet, verbraucht BLE weniger Strom, als das klassische Bluetooth. So ist es möglich, dass Geräte mit einer Knopfbatterie für Monate oder sogar Jahre betrieben werden können, ohne dass diese erneuert werden muss [7]. So schätzt die SIG, dass der durchschnittliche Stromverbrauch von BLE gegenüber ANT und ZigBee, welche ebenfalls als Low-Power-Funktechnologien gelten, um den Faktor 1,5 - 2 geringer ist [10].

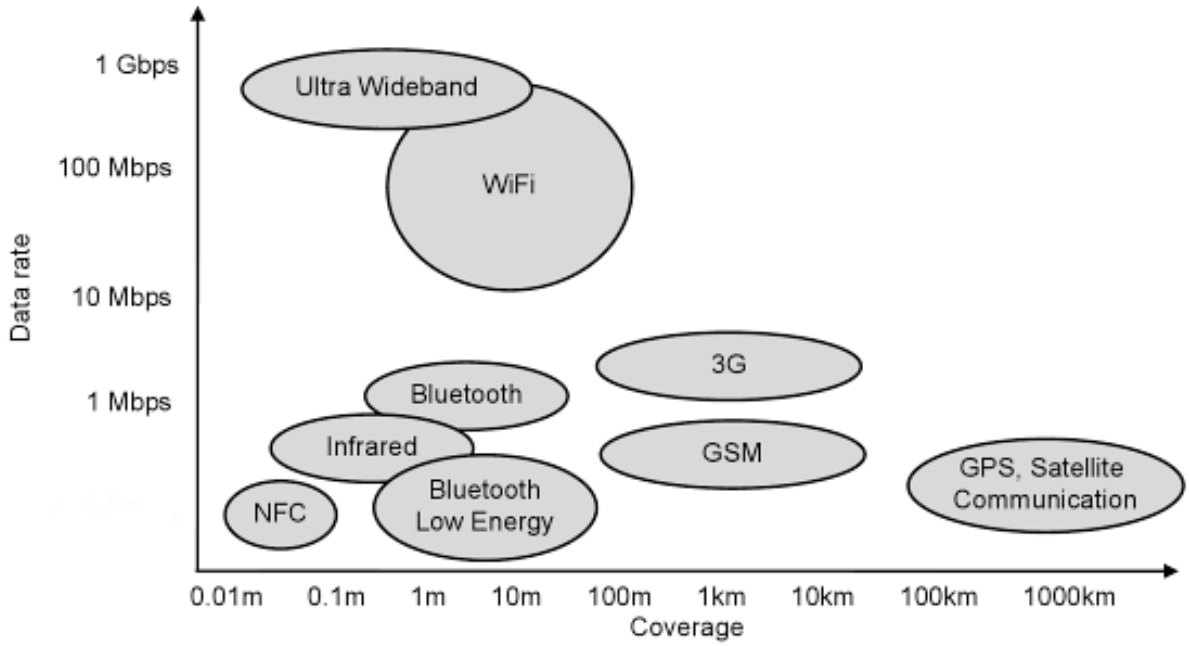


Abbildung 1: Überblick der Datenübertragungsrate und Reichweite verschiedener, kabelloser Technologien [7]

Die Abbildung 1 zeigt einen Überblick der Datenübertragungsraten und Reichweiten gängiger kabelloser Technologien zur Datenübertragung. Im Vergleich der Reichweiten liegt BLE gegenüber Bluetooth, trotz geringerem Energieverbrauch, etwas voraus. Die Übertragungsrate bei Bluetooth ist jedoch deutlich höher, sodass sich BLE mit einer Übertragungsrate von nur 1 MBit/s vor allem zum Übertragen kleiner Datenmengen, wie Sensordaten, eignet. Dies ist auch der Grund, warum BLE Bluetooth nicht als Standard zur Übertragung größerer Datenmengen, wie Audio- oder Videodateien, ablöst. Geräte wie Smartphones und Tablets mit Bluetooth 4.0 unterstützen sowohl weiterhin die Übertragungsraten BR, EDR und LE. Diese Geräte werden als Bluetooth Smart Ready bezeichnet und können

sich auch mit dem „klassischen“ Bluetooth 3.0 oder niedriger verbinden. Während z.B. Sensoren oder iBeacons, die lediglich kleine Datenmengen übertragen und möglichst wenig Strom verbrauchen sollen, als Bluetooth Smart Geräte bezeichnet werden und nur die Übertragungsrate von LE unterstützen. Diese Geräte können sich nur mit einem Bluetooth 4.0 oder einem höheren Chip verbinden [7, 11]. Durch die effiziente Übertragung von Daten ist mit BLE ein weiterer wichtiger Schritt in Richtung mobiles Bezahlen, sowie dem Internet der Dinge gesetzt. Der Begriff Internet der Dinge beschreibt, dass nahezu jedes Gerät mit dem Internet verbunden ist, um dem Benutzer Arbeit zu erleichtern oder ihm Informationen zu übermitteln. Ein Beispiel wäre ein Getränkekrug, der mit einem iBeacon versehen ist und die aktuelle Füllmenge misst. Diese Daten werden an ein Smartphone weitergeleitet, welches bei einer zu hohen Füllmenge den Benutzer darauf hinweist, dass dieser zu wenig Flüssigkeit zu sich nimmt.

2.3 iBeacon

Mit iBeacon hat Apple, wie auch PayPal mit dem PayPal Beacon, unter Nutzung der Bluetooth Low Energy-Technologie einen eigenen Standard zur Indoor-Positionsbestimmung veröffentlicht. Die iBeacon-Technologie ermöglicht es einem Endgerät, z.B. einem Smartphone oder einem Tablet, ein Signal zu empfangen, welches von einem kompatiblen Beacon innerhalb einer Reichweite von bis zu 100 m gesendet wird [12]. Die maximale Reichweite ist dabei abhängig von der Sendeleistung des Beacons.
Diese senden in regelmäßigen Abständen unter Verwendung des BLE-Standards drei Identifikationsnummern, welche von der dazugehörigen Applikation (App) auf einem Endgerät empfangen werden. Weitere Daten werden durch den iBeacon-Standard nicht übermittelt.
Anhand dieser Identifikationsnummern kann die App ein zugehöriges iBeacon mit bekanntem Standort eindeutig identifizieren. Dadurch kann auf die Position des Anwenders zurückgeschlossen werden, sodass dieser ortsspezifische Informationen erhält.
Die drei Identifikationsnummern bestehen aus einem 128 Bit Value-Wert, dem Universally Unique Identifier (UUID), sowie zwei 16 Bit Integer-Werten und der major- und der minor-Kennung [13].
Die UUID beschreibt alle iBeacons einer Organisation. Durch die major-Kennung können Gruppen von zusammengehörigen iBeacons mit derselben UUID beschrieben werden. Die minor-Kennung ermöglicht die Differenzierung von iBeacons in diesen Gruppen mit gleicher UUID und major-Kennung.
Möchte z.B. ein Krankenhaus jede Station mit der iBeacon-Technologie ausstatten, steht die UUID für das Krankenhaus. Hierbei besitzen alle iBeacons innerhalb des Krankenhauses dieselbe UUID. Jede Station bekommt eine eindeutige major-Kennung zugewiesen. Diese ist bei allen iBeacons auf der Station gleich. Innerhalb der Station können die iBeacons mit Hilfe der minor-Kennung auseinander gehalten werden.

Neben der Identifikation eines iBeacons besteht ein weiterer Vorteil der Technologie darin, dass der Abstand zum Smartphone über den proximity-Wert in vier Kategorien (Unknown, Immediate, Near oder Far) klassifiziert wird [3]. Dies geschieht durch einen von Apple vorgegebenen Algorithmus. Dabei wird die Genauigkeit über den accuracy-Wert angegeben. Eine Abstandsmessung über die Empfangsfeldstärke (RSSI) ist zwar ebenfalls über Bluetooth mit und ohne iBeacon möglich, erfordert jedoch einen vom Programmierer entwickelten Algorithmus, welcher die RSSI in eine relative Distanz umrechnet und mögliche Signaldämmer ausgleicht.

Unterstützt wird iBeacon ab iOS 7 und Android 4.3, sofern die Geräte einen BLE-Chip integriert haben. Dieser ist in den Apple-Produkten ab dem iPhone 4S, dem iPad 3 und dem iPod Touch 5, sowie in aktuellen Android-Geräten vorhanden.
Die Apple-Produkte können iBeacon Signale empfangen und senden, während die Android-Geräte nur Signale empfangen können.

2.4 Indoor-Navigation

2.4.1 Lateration am Beispiel von iBeacon

Die Lateration ist eine auf Abstandsmessung basierende Möglichkeit der Positionsbestimmung.
Bei diesem Verfahren muss die Position des festinstallierten, sendenden Objektes und der Abstand zu einem empfangenden Objekt bekannt sein. Im Rahmen dieser Arbeit soll eine Indoor-Navigation entstehen, mit der ein Smartphone basierend auf der iBeacon-Technologie geortet werden kann. Das jeweilige iBeacon ist dabei das sendende Objekt. Seine Position ist bekannt und fest, sodass sie in ein kartesisches Koordinatensystem übertragen werden kann. Das zu ortende Endgerät ist das empfangende Objekt, welches sich frei in einem Raum bewegen kann. Über die RSSI oder den proximity-Wert ist eine Abstandsmessung zwischen iBeacon und Smartphone möglich. Andere Technologien, wie GPS, nutzen zur Abstandsmessung nicht die Empfangsfeldstärke, sondern die Signallaufzeit. Nach welcher Methode die Abstandsmessung zwischen Endgerät und iBeacon erfolgt, ist für das Verfahren der Lateration nicht relevant. Das Verfahren kann sowohl bei der Navigation über GPS, wie auch bei einer iBeacon basierten Navigation eingesetzt werden. Der gemessene Abstand sollte jedoch möglichst genau sein.

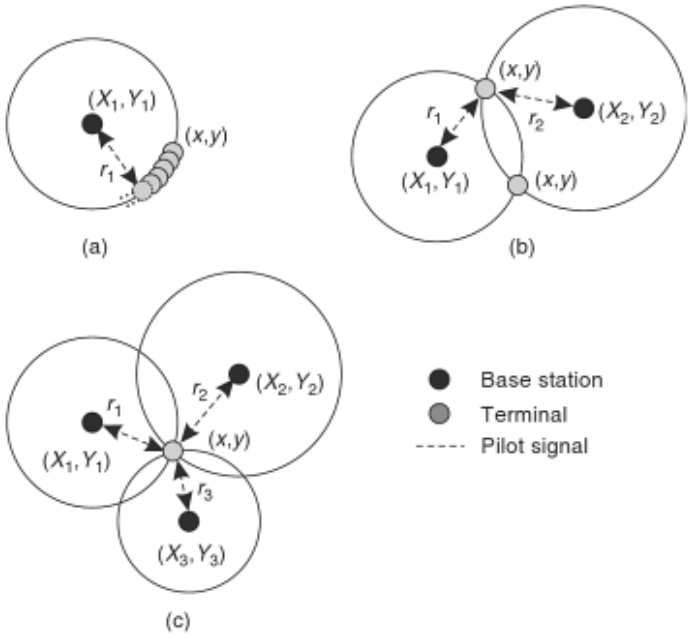


Abbildung 2: Lateration in 2-D [14]

In der Abbildung 2 wird die Lateration in einem zweidimensionalen Raum dargestellt. Der schwarze Kreis (Base station) symbolisiert ein iBeacon. Der graue Kreis (Terminal) stellt eine mögliche Position des Smartphones

dar. Die gestrichelte Linie (Pilot signal) steht für den Abstand zwischen iBeacon und Smartphone.
In der Abbildung 2(a) sind die Position des iBeacons (X_1,Y_1) und der Abstand (r_1) zum Smartphone (x,y) bekannt. Gesucht wird die Position des Smartphones (x,y). Wie in der Abbildung 2(a) zu sehen, kann sich diese auf jedem beliebigen Punkt des Kreises (Mittelpunkt: X_1, Y_1 und Radius: r_1) befinden. Die Funktionsgleichungen dazu lauten: $y= Y \pm \sqrt{r^2 - (x-X)^2}$ bzw. $x= X \pm \sqrt{r^2 - (y-Y)^2}$. Um die mögliche Position des Smartphones weiter einzugrenzen, kann ein zweiter iBeacon hinzugezogen werden. Um diesen lässt sich, wie in Abbildung 2(b) zu sehen, ein weiterer Kreis mit Radius r_2 bilden. Die beiden Kreise bilden nun zwei Schnittpunkte, sofern gilt: (X_1, Y_1) ≠ (X_2,Y_2). Dadurch kann die mögliche Position des Smartphones auf diese beiden Schnittpunkte reduziert werden. Kann keiner dieser Positionen durch eine logische Schlussfolgerung ausgeschlossen werden, etwa weil beide Punkte innerhalb des definierten Raumes liegen, muss ein dritter iBeacon zur genauen Positionsbestimmung hinzugefügt werden. Hierbei spricht man von einer Trilateration, welche in Abbildung 2(c) zu sehen ist. Die drei Kreise der iBeacons bilden nur einen gemeinsamen Schnittpunkt. Dies ist die Position des Smartphones. Zur Bestimmung der Position eines Smartphones in einem zweidimensionalen Raum werden mit Hilfe der iBeacon-Technologie demnach mindestens drei iBeacons benötigt. Dabei gilt: je genauer die Abstände r_1, r_2 und r_3 der iBeacons zum Smartphone und je genauer die Positionen [(X_1, Y_1); (X_2, Y_2); (X_3, Y_3)] der iBeacons, desto genauer ist die ermittelte Position (x, y) des Smartphones.

Dieses Verfahren lässt sich auch auf den dreidimensionalen Raum erweitern. In der Abbildung 3 wird dies dargestellt. Anstatt eines Kreises bildet sich im dreidimensionalen Raum eine Kugel mit dem Radius r um das iBeacon. Das Smartphone kann sich theoretisch auf der gesamten Oberfläche der Kugel befinden.

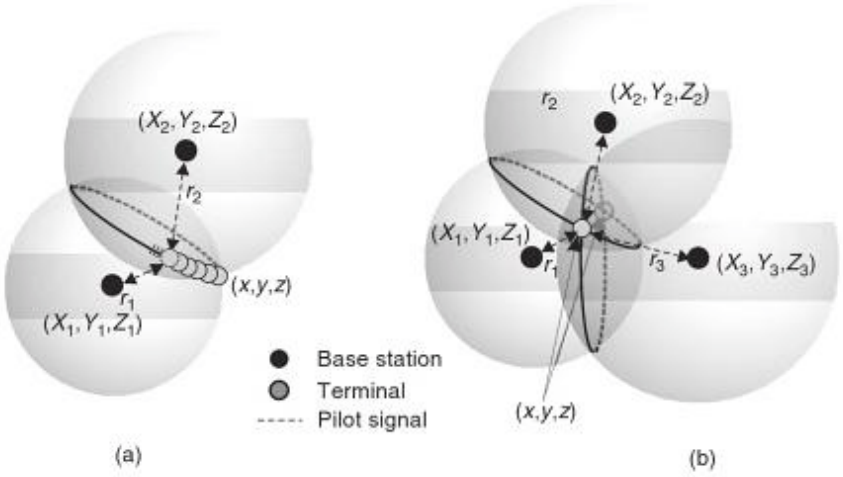


Abbildung 3: Lateration im dreidimensionalen Raum [14]

Um die mögliche Position weiter einzugrenzen kann, wie in Abbildung 3(a), ein zweites iBeacon hinzugefügt werden. Es entstehen zwei Kugeln, die einen gemeinsamen Schnittkreis bilden. Dargestellt wird dieser durch das Pilot signal und allen Terminals mit den Koordinaten (x, y, z). Eine Terminal Position ist dabei gleich der gesuchten Position des Smartphones.
Um die Anzahl der möglichen Terminals weiter einzugrenzen, wird wie in Abbildung 3(b) ein drittes iBeacon hinzugefügt. Die Anzahl der Terminals und damit die möglichen Positionen des Smartphones, werden so auf die zwei Schnittpunkte reduziert. Kann keiner der zwei Terminals durch eine logische Schlussfolgerung ausgeschlossen werden, reichen drei iBeacons im dreidimensionalen Raum nicht aus, um die Position eines Smartphones eindeutig zu bestimmen. Unter der Hinzunahme eines vierten iBeacons ist jedoch die genaue Positionierung eines Smartphones im dreidimensionalen Raum möglich. Auch hier gilt: je genauer die Abstände r_1, r_2, r_3 und r_4 der iBeacons zum Smartphone und je genauer die Positionen $[(X_1, Y_1); (X_2, Y_2); (X_3, Y_3); (X_4, Y_4)]$ der iBeacons, desto genauer die Position (x, y, z) des Smartphones.

2.4.2 Fingerprinting-Verfahren am Beispiel iBeacon

Eine weitere Möglichkeit zur Positionsbestimmung in Gebäuden ist das Vorgehen nach dem Fingerprinting-Verfahren.

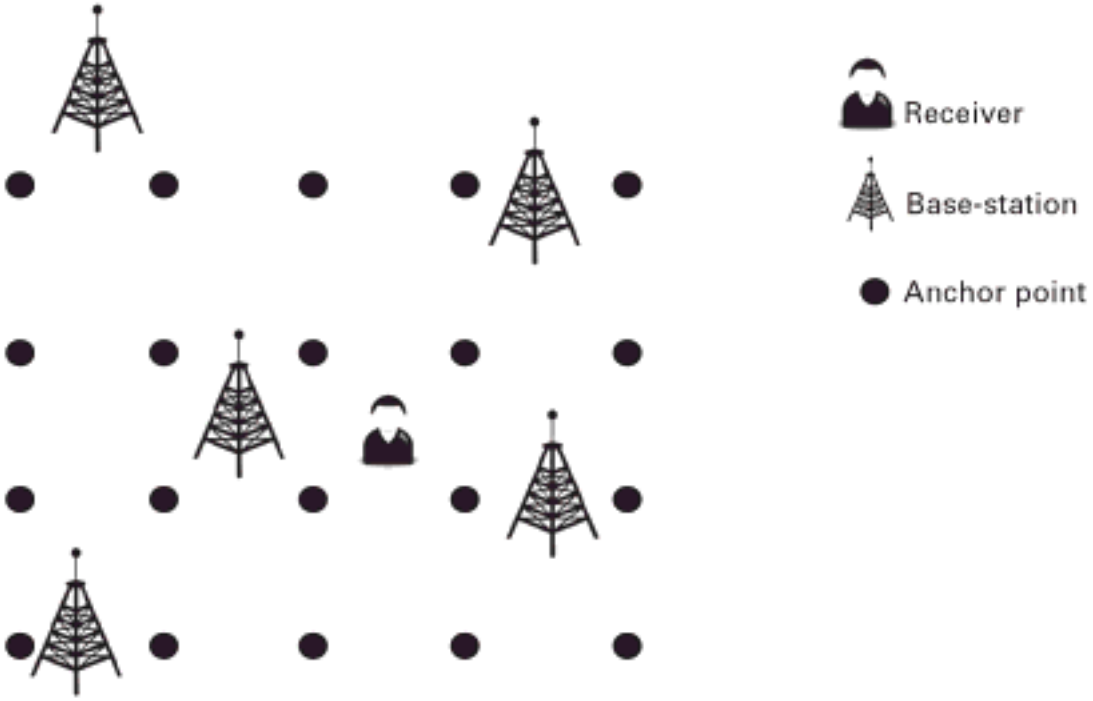


Abbildung 4: Radio Map des Fingerprinting-Verfahrens [15]

Bei diesem Verfahren werden die iBeacons (Base-station) im Raum verteilt, dabei muss ihre Position nicht bekannt sein. Um die Position des Endgeräts (Receiver) zu ermitteln, ist zuerst eine Trainingsphase notwenig, in der die sogenannte Radio Map erstellt wird.
Dazu erfolgt vor der eigentlichen Positionsbestimmung (online-Phase) eine Messung der Abstände zwischen den verschiedenen Anchor points zu den im Raum verteilten iBeacons. Die gemessenen Abstände und die Position der Anchor points werden zu einem Kalibrierungstupel zusammengefasst und in einer Datenbank gespeichert.
Soll nun eine Positionsbestimmung (online-Phase) durchgeführt werden, werden die Abstände vom Endgerät (Receiver) zu allen iBeacons (Base-stations) gemessen. Die Abstände werden zu einem Ortungstupel zusammengefasst. Dieses Ortungstupel wird mit allen Kalibrierungstupeln der Radio Map verglichen. Das ähnlichste bestimmt dabei den Standort des Endgerätes. Als Ähnlichkeitsmaß können verschiedene Methoden wie das Euklidisch-Verfahren oder eine statistische Bayes-Methode verwendet werden [30].

Folgende Formel beschreibt dabei das Vorgehen des Euklidischen Verfahrens:

$$d(k,o) = \sqrt{(k_1\text{-}o_1)^2 + (k_2\text{-}o_2)^2 + (k_3\text{-}o_3)^2}$$

Kalibrierungstupel: k; Ortungstupel: o

Zusätzlich können durch das Verfahren der Interpolation alle Positionen auf der Radio Map, die keine Anchor points sind, errechnet werden. Ebenfalls ist es denkbar, die Radio Map durch maschinelles Lernen in der online-Phase zu erweitern.
Vorteile des Fingerprinting-Verfahren sind zum einen, dass die Schwankungen der einzelnen iBeacons untereinander bei der Abstandsmessung berücksichtigt werden. Zum anderen werden ortsbezogene Störquellen, wie Wände oder W-Lan Netzwerke in die Positionsbestimmung mit einbezogen.
Als Nachteil ist aufzuführen, dass mobile Störquellen, wie Personen, bei nur einer Radio Map nicht berücksichtigt werden. Es ist jedoch denkbar, mehrere Radio Maps zu erstellen z.B. von einem Raum ohne Personen und demselben Raum gefüllt mit Personen. Über die Kamera des Endgeräts kann automatisch ermittelt werden, ob sich Personen im Raum befinden.

3 Marktanalyse

3.1 Marktanalyse der Hardware

Zu Beginn der Arbeit befinden sind noch keine offiziellen iBeacon-Sender auf dem Markt. Dies dürfte sich im Laufe der Arbeit ändern, da Apple im Februar 2014 die iBeacon-Spezifikation für eingeschriebene Unternehmen des Apple Made for iPhone/iPod/iPad-Programms (MFi)[3] zugänglich gemacht hat. Dadurch haben externe Hersteller, nach einer erfolgreichen Lizenzierung für das MFi-Programm, die Möglichkeit iBeacons herzustellen und diese zertifizieren zu lassen. Zudem kann das Produkt mit dem MFi-Logo aus Abbildung 5 und dem Namen iBeacon beworben werden.

Abbildung 5: Made for iPhone Logo [16]

Auch nach Veröffentlichung der Spezifikation ist nicht viel über dessen Inhalt bekannt, da die Unternehmen bei der Beantragung der Lizenzierung einem Non Disclosure Agreement (NDA), also einer Geheimhaltungserklärung zustimmen müssen [16]. Aber auch ohne Lizenzierung für das MFi-Programm haben Hersteller die Möglichkeit, BLE-Sender herzustellen, welche als iBeacon fungieren können. So hat das Unternehmen Estimote[4] zwar mittlerweile eine Lizenzierung erhalten, produzierte und verkaufte jedoch bereits vor der Veröffentlichung der iBeacon-Spezifikation ihre Beacon-Sender. Dennoch ist davon auszugehen, dass durch die Veröffentlichung der Spezifikation und dem wachsendem Interesse an iBeacon weitere Produkte auf dem Markt erscheinen und dadurch der Preis fallen wird.

[3] www.mfi.apple.com

[4] www.estimote.com

3.1.1 Estimote Beacon

Das im Jahre 2012 gegründete Unternehmen Estimote Inc. vertreibt Beacons, welche mit der iBeacon-Technologie kompatibel sind. Zurzeit können auf der Internetseite des Unternehmens drei Beacons für 99$ bestellt werden [17].

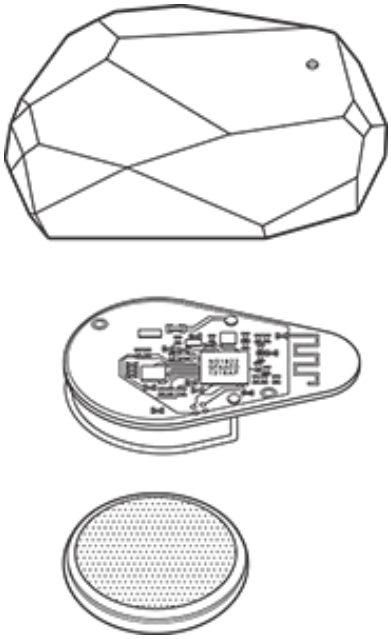


Abbildung 6: Aufbau eines Estimote Beacons [18]

Die Abbildung 6 zeigt den Aufbau eines Estimote Beacons. Die Sender haben eine maximale Reichweite von 70 m und sind ausgestattet mit einem 32-bit ARM® Cortex M0 Prozessor, 256kB Flash-Speicher, einem 2,4 GHz Bluetooth 4.0 Chip, einem Beschleunigungs-, sowie einem Temperatursensor. Als Stromquelle dient eine Knopfbatterie, welche laut Hersteller über 2 Jahre ausreichend Energie liefert. Diese Zeit ist jedoch stark abhängig von der Regelung der Sendeleistung (TxPower) und damit der Reichweite des Signals, sowie dem Intervall, in denen das Signal versendet wird. Aus diesen Gründen hat der Benutzer, wie in Abbildung 7 zu sehen, die Möglichkeit, beide Einstellungen anzupassen.

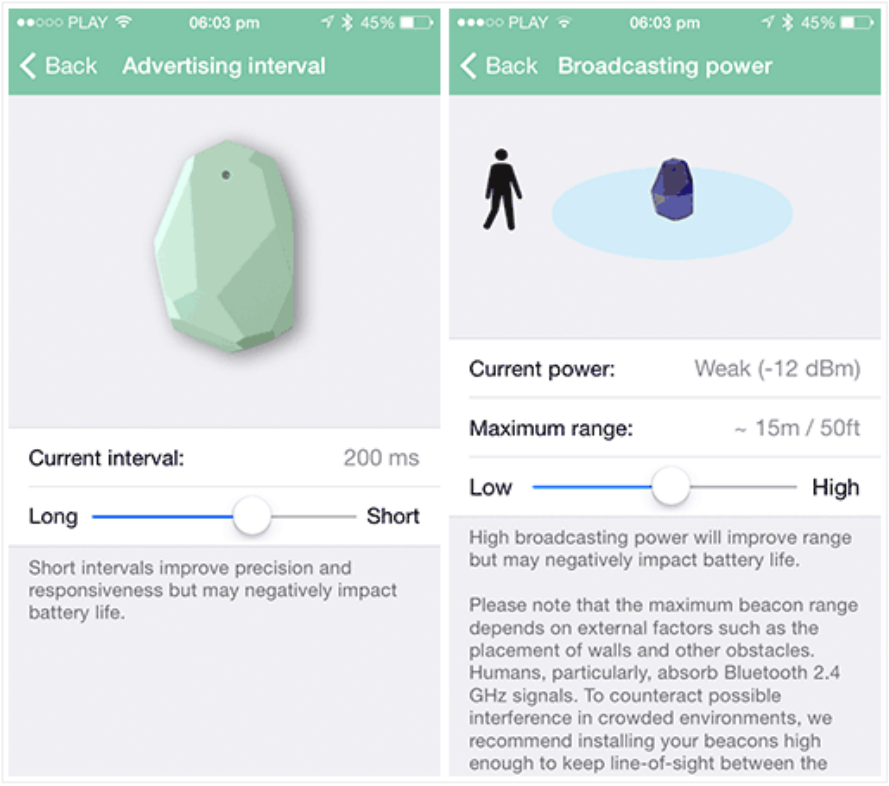


Abbildung 7: Einstellungen für das Intervall und die TxPower [17]

Die Abstände lassen sich dabei zwischen 1x pro Sekunde (1Hz) und 10x pro Sekunde (10Hz) justieren. Eine höhere Frequenz sorgt dabei für ein stabileres Signal und eine schnellere Reaktion der App, jedoch auch für einen höheren Stromverbrauch.
Der Abstand zwischen dem Endgerät und einem Estimote Beacon wird über die empfangende RSSI und der Measured Power errechnet. Die Measured Power ist durch Estimote vorgegeben und kann vom Benutzer nicht verändert werden. Sie wird neben den drei Identifikationsnummern durch das Beacon versendet. Die Measured Power wird ermittelt, indem die empfangende RSSI, gesendet von einem Beacon, welches in 1 m Abstand zum Endgerät installiert ist, gemessen wird. Die Ermittlung der Measured Power ist bei den Estimote Beacons erforderlich, da die TxPower vom Benutzer eingestellt werden kann [19].

3.1.2 BlueBar Beacon

Das Unternehmen Blue Sense Networks[5] hat mit BlueBar Beacon ebenfalls iBeacon kompatible Beacons entwickelt. Vertrieben werden sie in den drei Variationen Integration Kit, Retail und USB (Abbildung 8).

Abbildung 8: Produktportfolio Blue Sense Network [12]

Die BlueBar Beacons haben eine Reichweite von 30 - 100 m und eine einstellbare Sendeleistung von -24dBm bis 0dBm. Der Anwender hat die Möglichkeit bei allen Varianten der BlueBar Beacons die UUID, die major-, die minor-Kennung, die Sendeleistung, sowie das Sendeintervall zu konfigurieren.
Der Modelltyp Retail ist baugleich mit der BlueBar Beacon Integration, jedoch zusätzlich in ein Kunststoffgehäuse umhüllt. Beide BlueBar Beacons sind mit einer Knopfbatterie (CR2450 3V) ausgestattet, welche laut Herstellerangaben bis zu zwei Jahre ausreichend Strom liefert. Der BlueBar Beacon USB wird hingegen über einen USB-Anschluss mit Energie versorgt. Dies schränkt ihn, wie den PayPal Beacon, in der Mobilität ein, erspart aber den Austausch einer Batterie.

[5] www.bluesensenetworks.com

3.1.3 PayPal Beacon

PayPal[6] arbeitet zurzeit ebenfalls daran, eigene Beacons (Abbildung 9) auf den Markt zu bringen. Zurzeit ist ein Start der PayPal Beacon in den USA, UK, Australien, Hong Kong und Japan für Anfang 2014 [20], im zweiten Halbjahr dann auch für Europa geplant [21].

Abbildung 9: PayPal Beacon [20]

Die PayPal Beacon basieren, wie iBeacon-Sender, ebenfalls auf der Bluetooth Low Energy - Technologie. Es ist geplant, dass die API für Entwickler freigegeben wird, damit die Technologie in externe Apps integriert werden kann. Im Vordergrund von PayPal Beacon steht das mobile Bezahlen. Dafür reicht die Abdeckung mit der PayPal Beacon - Technologie am Kassenbereich eines Ladens in der Regel aus. Der geringe Bedarf an einer großen Abdeckung mit PayPal Beacon ermöglicht es, die Beacon mit einem USB-Anschluss auszustatten. Im Kassenbereich sind häufig Computer mit USB-Anschluss vorhanden, die die PayPal Beacon mit Strom versorgen können. Die festinstallierte Stromquelle ermöglicht es wiederum, den Sendebereich auszudehnen. Zur Übertragung ortsspezifischer Informationen oder der Indoor-Navigation innerhalb eines Gebäudes sind die PayPal Beacon jedoch weniger geeignet, da sie aufgrund des USB-Anschlusses nicht beliebig in einem Raum verteilt werden können.

[6] www.paypal.com

3.2 Marktanalyse der Software

Die Positionsbestimmung und damit die Indoor-Navigation mittels dem Verfahren der Lateration über Messung der RSSI ist auch ohne den Standard iBeacon nur mit BLE-Technologie möglich. Apple hat das Verfahren mit iBeacon jedoch stark vereinfacht. So wird die relative Distanz zwischen dem Sender und dem Empfänger automatisch über die RSSI ermittelt und in 3 Klassen eingeteilt.

Estimote hat für ihr Beacon eine SDK veröffentlicht, welche als Wrapper bzw. Schnittstelle zu Apples CoreLocation Framework fungiert [18].

3.3 Marktanalyse Indoor-Navigation

In diesem Unterkapitel soll untersucht werden, welche Alternativen der Indoor-Navigation zu iBeacon auf dem Markt existieren. Ebenfalls soll eine Marktanalyse zur Kombination dieser Technologien, der sogenannten integrierten Indoor-Navigation, durchgeführt werden.

3.3.1 QR-Code

Bei der Navigation über Quick Response (QR) - Codes wird der Raum mit eindeutigen und festinstallierten QR-Codes ausgestattet. Die QR-Codes können z.B. ausgedruckt und freizugänglich an die Wand gehängt werden. Der Benutzer hat mit seinem Smartphone und einem QR-Scanner nun die Möglichkeit, den Code über die Kamera einzulesen. Der QR-Code wird vom QR-Scanner entschlüsselt und die Informationen an eine App oder eine Webseite weitergeleitet. Die Informationen beinhalten die Position des QR-Codes, sodass dem Anwender ein Standort zugeordnet werden kann. Dieses System bietet die Vorteile, dass es kostengünstig installiert und betrieben werden kann. Einmal aufgehängte QR-Codes sind über mehrere Jahrzehnte haltbar. Des Weiteren sind im Gegenzug zur funkbasierten Navigation keine Signaldämmer vorhanden. Auch der Energieverbrauch des Smartphones wird gering gehalten, da Funkübertragungstechniken wie WLAN oder Bluetooth nicht benötigt werden. Allerdings muss der Benutzer einen QR-Scanner besitzen und für jede Aktualisierung des Standortes, also theoretisch bei jeder Abzweigung des Weges, einen neuen QR-Code scannen. Gerade durch die aufwendige Standortaktualisierung ist die Indoor-Navigation über QR-Codes nicht sinnvoll. Für die Übermittlung von einmaligen, kostengünstigen und standortbasierten Informationen kann das QR-Code Verfahren jedoch nützlich sein.

3.3.2 WLAN

Die Indoor-Navigation über WLAN erfolgt, wie die Navigation über iBeacon, ebenfalls über das in Kapitel 2.4.1 vorgestellte Verfahren der Lateration. Die Abstandsmessung zwischen dem sendenden Objekt und dem Endgerät erfolgt durch die Messung des RSSI-Wertes. Eine Indoor-Navigation über WLAN bietet gegenüber iBeacon den Vorteil, dass zurzeit deutlich mehr Endgeräte mit einem WLAN-, als mit einem Bluetooth 4.0 - Chip ausgestattet sind. Ein weiterer Vorteil gegenüber iBeacon ist die höhere Datenübertragungsrate von WLAN, sowie das häufige schon Vorhandensein von WLAN-Netzwerken in Gebäuden. Allerdings müssen, wie in Kapitel 2.3 festgestellt, bei der Indoor-Navigation kaum Daten übertragen werden. Somit ist es kein relevanter Vorteil, sondern vielmehr durch den höheren Energieverbrauch ein Nachteil. Während ein iBeacon über Jahre mit einer Knopfbatterie betrieben werden kann, benötigt der WLAN-Router durch den hohen Energieverbrauch in der Regel eine Versorgung aus einer fest installierten Stromquelle. Zwar muss dadurch die Stromquelle nicht erneuert werden, es entstehen jedoch höhere Energiekosten, sowie Kosten bei der Verlegung und Wartung der Stromkabel. Zudem reicht das Empfangen des Signals von einem bereits fest installierten WLAN-Router zur genauen Positionsbestimmung nicht aus. Es werden, wie bei der Navigation über iBeacon, drei bzw. vier sendende Objekte benötigt, um das Signal zu orten. Hier zeichnet sich ein weiterer Vorteil von iBeacon ab. Ein Bluetooth 4.0 Sender ist bereits jetzt schon günstiger, als ein WLAN-Router oder Repeater, zumal davon auszugehen ist, dass die Preise in den nächsten Jahren noch stark fallen werden.
Ein Problem, welches sich iBeacon und WLAN bei der Indoor-Navigation teilen, ist die Genauigkeit der Abstandsmessung beim Verfahren der Lateration. Bei beiden Methoden wird die RSSI durch Hindernisse reduziert.
Den einzigen Mehrwert den WLAN zurzeit bietet ist das häufigere Vorkommen eines WLAN-Chips in den Endgeräten. Aufgrund der Tatsache, dass sich Bluetooth bereits als Standard etabliert hat, ist davon auszugehen, dass die Zahl der Endgeräte, die Bluetooth 4.0 unterstützen, weiter steigen wird. Somit wird eine Indoor-Navigation über WLAN aufgrund der Nachteile gegenüber iBeacon für die Zukunft immer uninteressanter.

3.3.3 Satellitengestützte Navigationssysteme

Die in den USA entwickelte Technologie der GPS-Navigation hat sich innerhalb der Outdoor-Navigation bewährt. Auch die europäische Variante Galileo konnte 2013 die erste Positionsbestimmung eines Flugzeugs durchführen. Mit einem Start des Systems ist Ende 2014 zu rechnen [22]. Ebenfalls im Aufbau befindet sich das chinesische System Beidou. Das Verteidigungsministerium der Russischen Föderation hat mit dem Globalnaja nawigazionnaja sputnikowaja sistema (GLONASS) zudem ein ähnliches System entwickelt. Alle vier Systeme arbeiten satellitengestützt und haben den Vorteil, sofern die Satelliten verwendet werden dürfen, dass kein weiteres Satellitennetzwerk installiert werden muss. Das Erstellen und Warten eines eigenen Netzwerkes, wie bei iBeacon- oder WLAN-Systemen entfällt somit. Allerdings teilen GPS, Galileo, Beidou und GLONASS zwei gemeinsame Probleme. Zum einen ist die Ermittlung der Höhenmeter, also der Etage, des Endgerätes nicht möglich. Zum anderen kann die Positionsbestimmung bei herkömmlichen Endgeräten ungenau, nicht möglich oder sehr zeitaufwendig sein, da die elektromagnetischen Wellen der Satelliten beim Durchdringen eines Mediums gehemmt werden. Bei dem am häufigsten verwendeten System GPS reicht die Signalstärke dennoch selbst in einer Tiefgarage für die Positionsbestimmung noch aus. Die zusätzliche Dämmung muss jedoch mit mehr Rechenleistung des Empfängers ausgeglichen werden. Dies ist insbesondere im Low-Cost Bereich nicht der Fall, da hier viele Dezibel an Empfindlichkeit zugunsten einer billigen Lösung verschenkt werden [2]. Die Ermittlung einer Etage ist selbst mit viel Rechenleistung nicht möglich. Es können jedoch mehrere Systeme kombiniert werden. Dies geschieht z.B. beim Assisted Global Positioning System (AGPS), um die „Time to first fix“ (TTFF), also die Zeit der ersten Lokalisierung zu reduzieren. War das Endgerät mehr als 2 - 6 Stunden nicht aktiv, kann die Dauer der TTFF ca. 45 Sekunden betragen. War das Gerät sogar mehrere Tage mit keinem Satelliten verbunden, kann die TTFF bis zu 12,5 Minuten andauern [23]. Die Idee hinter der AGPS-Technologie ist es, die Anzahl der geeigneten Satelliten zu reduzieren. Dies geschieht z.B. durch die Identifikation der Mobilfunkzelle oder der Ephemeriden. Letzteres ist eine Liste mit den genauen Positionen eines Himmelskörpers als eine Funktion der Zeit. Die Daten können dabei in Echtzeit oder nachberechnet vorliegen [23].
Mit einem leistungsstarken Endgerät kann GPS also zur Indoor-Navigation verwendet werden, sofern die Höhenmeter irrelevant sind. Der Vorteil liegt vor allen darin, dass anders als bei iBeacon kein zusätzliches Netzwerk aufgebaut oder gewartet werden muss. Dafür können die Empfänger, je nach Signaldämmung des Gebäudes, sehr kostenintensiv sein. Zudem ist

nicht davon auszugehen, dass in einem neuen Smartphone ein leistungsstarker GPS-Chip für die Indoor-Navigation installiert ist. Die Installation eines BLE-Chips gilt jedoch als wahrscheinlich.

3.3.4 Integrierte Indoor-Navigation

Bei integrierten Navigationssystemen werden verschiedene Navigationsverfahren und Sensortasten kombiniert. Das Ziel hierbei ist es, die Nachteile eines einzelnen Navigationsverfahren durch die Vorteile eines anderen Navigationsverfahren auszubessern, um so eine möglichst genaue und schnelle Navigation zu ermöglichen [24]. Bei einem modernem Smartphone können verschiedene Sensoren und Empfangsfeldstärken wie GSM, 3G/4G (LTE), WLAN, Kompass, Barometer, Beschleunigungssensor, Gyroskop, Bluetooth und GPS ausgewertet werden [25].

4 Test der Estimote Beacons

Um zu überprüfen, ob iBeacon zur Indoor-Navigation geeignet ist, werden Abstandsmessungen aus verschieden Entfernungen mit diversen Störquellen durchgeführt.
Als Empfangsgerät wird zuerst ein iPhone 5s (Model A1457) mit iOS 7.1.1 verwendet, ab der Versuchsreihe *4.2 Einfluss von Signaldämpfern* ist iOS 7.1.2 auf dem iPhone installiert. Im weiteren Verlauf der Versuchsreihe wird zusätzlich ein iPad 4 mit iOS 7.1.2 hinzugezogen. Zum Senden werden drei Estimote Beacons in der Hardware Version D3.4 und der Firmware A2.1 benutzt. Die Testsoftware wird in Objective-C mit der Estimote SDK 2.0.0 entwickelt.
Die Beacons wurden dabei vertikal mit dem Punkt nach oben platziert, wie es von Estimote empfohlen wird [26]. Dies ist in Abbildung 10 zu sehen.

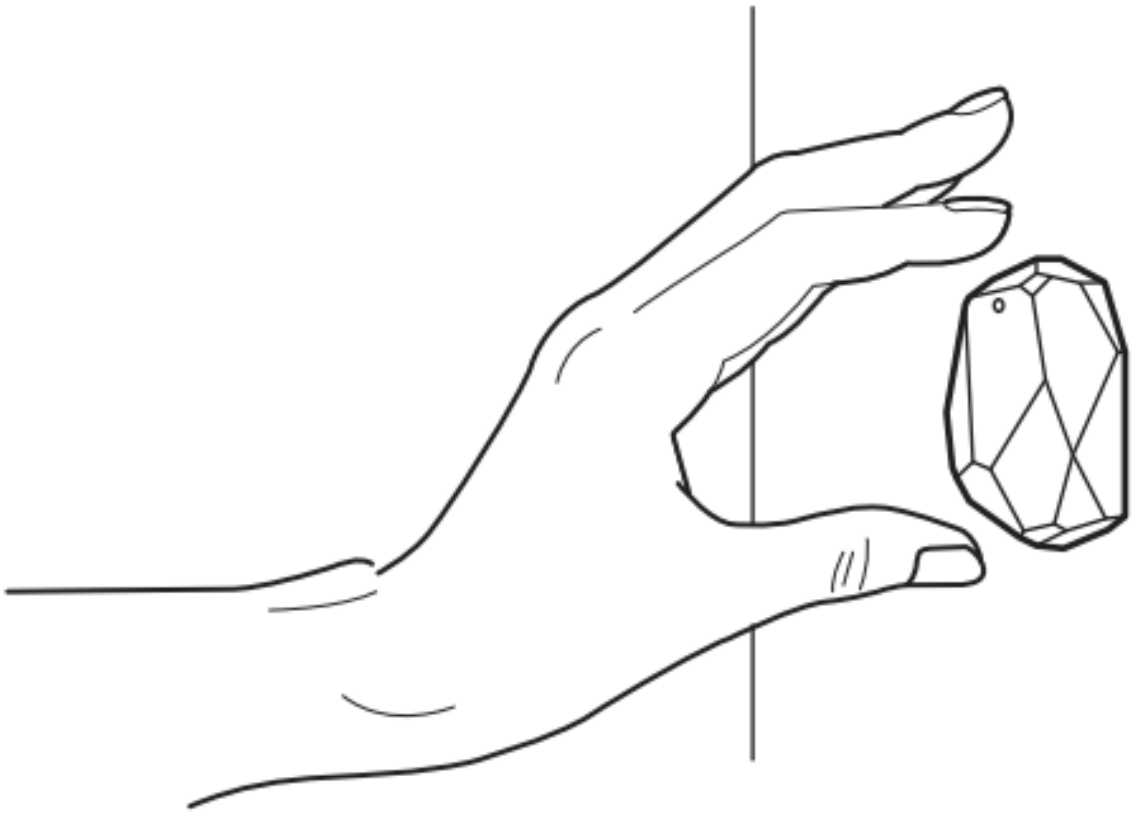


Abbildung 10: Optimale Platzierung der Estimote Beacons [26]

Laut einem Kommentar von Wojtek Borowicz, einem Mitarbeiter der Estimote Inc., zu dem Artikel RSSI, Range, Zones, and Distance Accuracy spielt die Interferenz zwischen einzelnen Beacons bei gleichzeitiger Benutzung keine signifikante Rolle [27]. Aus diesem Grund und um einen direkten Vergleich zwischen den einzelnen Beacons herzustellen, wird die RSSI von allen drei Beacons gleichzeitig durch ein iPhone gemessen.
Die errechneten Mittelwerte in den Tabellen ergeben sich aus den Beträgen der Werte in den Spalten bzw. Zeilen.

4.1 Grundlegende Versuche

In diesem Kapitel werden grundlegende Versuche mit den Estimote Beacons durchgeführt. Folgende Kriterien sollen dabei untersucht werden:

1) Wie viele Messungen sind erforderlich, um Schwankungen bei der Abstandsmessung zwischen iPhone und Beacon auszugleichen? Wie viele Messungen sind im Rahmen der Positionsbestimmung zeitlich vertretbar?

2) Liefert der Mittelwert oder der Median genauere Ergebnisse?

3) In welcher Position liefern die Beacons den genausten Abstand?

4) Laut Hersteller ist die gemessene Distanz präziser, wenn die Sendeleistung (TxPower) der Beacons möglichst hoch und das Sendeintervall möglichst gering ist [28]. Kann dies im Feldversuch bestätigt werden?

5) Wie präzise sind die Messungen unter optimalen Bedingungen?

4.1.1 Anzahl der Messungen

Versuchsbeschreibung:

In diesem Versuch wird untersucht, welche Anzahl von Messungen das effizienteste Ergebnis liefert. Dazu werden der jeweilige Mittelwert, Median und die Varianz von 1, 10, 20 und 100 Abstandsmessungen miteinander verglichen. Die gemessenen Werte werden jeweils nur für einen Versuch verwendet. Für den ersten Versuch werden also nicht die ersten 10 Werte der 100 Messungen verwendet. Lediglich für den Versuch mit nur einer Messung wird der erste Wert aus den 10 Messungen verwendet.

Versuchsaufbau:

Das iPhone und die drei Estimote Beacons sind dazu auf einer Höhe von 85 cm gelagert. Die Sendeleistung des Beacons beträgt 4 dBm, das Sendeintervall 100 ms. Der Abstand vom iPhone zu den Beacons beträgt in der ersten Versuchsreihe 5 m und in der zweiten Versuchsreihe 8 m.

Folgende Abbildung skizziert den Versuchsaufbau:

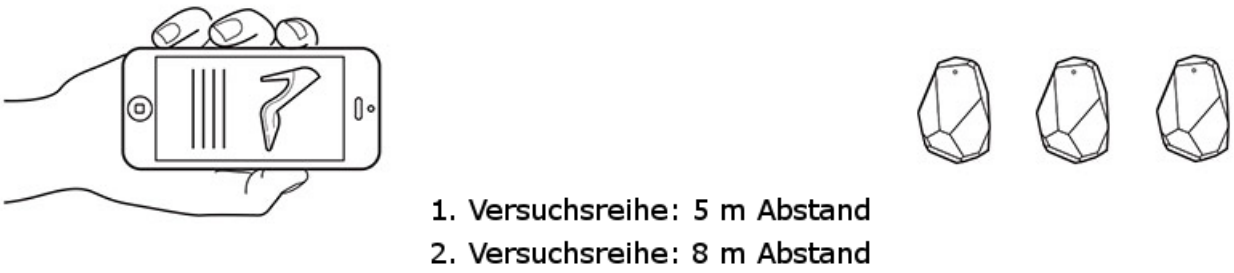


Abbildung 11: Versuchsaufbau des Versuchs 4.1.1
verwendete Grafiken von www.estimote.com

Versuchsergebnis:

4.1.1: minimal gemessener Wert auf einer Distanz von 5

	Beacon 1	Beacon 2	Beacon 3
1 Messung	1	1,14	0,01
10 Messungen	1	1,14	0,01
20 Messungen	1,11	1,29	0,01
100 Messungen	1,14	1,29	0,01

4.1.1: maximal gemessener Wert auf einer Distanz von 5

	Beacon 1	Beacon 2	Beacon 3
1 Messung	1	1,14	0,01
10 Messungen	1,14	1,29	0,01
20 Messungen	1,14	1,33	0,01
100 Messungen	1,30	1,45	0,02

4.1.1: Varianz der gemessenen Werte auf einer Distanz

	Beacon 1	Beacon 2	Beacon 3
10 Messungen	0,002	0,002	0
20 Messungen	0	0	0
100 Messungen	0,001	0,001	0

4.1.1: (Median der gemessenen Distanzen) - (wirkliche Distanz 5 m)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
1 Messung	-4,00	-3,86	-4,99	4,28
10 Messungen	-3,87	-3,72	-4,99	4,19
20 Messungen	-3,86	-3,68	-4,99	4,17
100 Messungen	-3,72	-3,65	-4,99	4,12
Mittelwert der Absolutwerte	3,86	3,72	4,99	

4.1.1: minimal gemessener Wert auf einer Distanz von 8 m

	Beacon 1	Beacon 2	Beacon 3
1 Messung	1,67	1,67	0,02
10 Messungen	1,67	1,67	0,02
20 Messungen	2,07	2,19	0,02
100 Messungen	2,00	2,17	0,02

4.1.1: maximaler gemessener Wert auf einer Distanz von 8 m

	Beacon 1	Beacon 2	Beacon 3
1 Messung	1,67	1,67	0,02
10 Messungen	2,29	1,93	0,03
20 Messungen	2,28	2,45	0,03
100 Messungen	2,45	2,50	0,03

4.1.1: Varianz der gemessenen Werte auf einer Distanz von 8 m

	Beacon 1	Beacon 2	Beacon 3
10 Messungen	0,042	0,013	0
20 Messungen	0,005	0,005	0
100 Messungen	0,009	0,007	0

4.1.1: (Mittelwert der gemessenen Distanzen) - (wirkliche Distanz 8 m)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
1 Messung	-6,33	-6,33	-7,98	6,88
10 Messungen	-5,91	-6,18	-7,97	6,68
20 Messungen	-5,81	-5,16	-7,97	6,31
100 Messungen	-5,83	-5,63	-7,97	6,47
Mittelwert der Absolutwerte	5,97	5,82	7,97	

4.1.1: (Median der gemessenen Distanzen) - (wirkliche Distanz 8 m)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
1 Messung	-6,33	-6,33	-7,98	6,88
10 Messungen	-5,77	-6,09	-7,97	6,61
20 Messungen	-5,79	-5,73	-7,97	6,49
100 Messungen	-5,85	-5,61	-7,97	6,47
Mittelwert der Absolutwerte	5,93	5,94	7,97	

Versuchsergebnis:

Es fällt auf, dass die gemessene Distanz der Beacons sehr stark von der wirklichen Distanz abweicht. Besonders auffällig ist, dass der dritte Beacon eine sehr niedrige Distanz ermittelt und dadurch starke Abweichungen von der wirklichen Distanz entstehen. Ebenfalls ist auffällig, dass alle Distanzen zu niedrig ermittelt werden. Dies könnte an den optimalen Bedingungen der Messung und dem eingebauten Entstörungsalgorithmus liegen [27].

Aus diesem Grund werden in späteren Versuchen Messungen mit Störquellen durchgeführt.

Die genausten Ergebnisse werden bei 100 Messungen durch Ermittlung des Median erzielt. Die Messung für drei Beacons dauert jedoch 1:40 min. Dies ist zur Positionsbestimmung zu lange, da sich der Anwender bei der Navigation mit dem Smartphone bewegt. Aufgrund dessen, dass die Messung für 20 Werte 20 Sekunden benötigt und die Dauer der Messung für 10 Werte nur 7 Sekunden benötigt und sich die Ergebnisse für die Positionsbestimmung nicht relevant voneinander unterscheiden, empfiehlt es sich, zur Positionsbestimmung 10 Messwerte oder weniger auszuwerten. Von der Auswertung nur eines Messwertes ist abzusehen, da dieser nicht durch andere Messwerte bestätigt werden kann.

4.1.2 Mittelwert oder Median

Versuchsbeschreibung:

In diesem Versuch wird untersucht, ob der Mittelwert oder Median genauere Ergebnisse liefert.

Versuchsdurchführung:

Dieser Versuch unterscheidet sich vom Versuchsaufbau nicht vom vorherigen Versuch, sodass diese Werte übernommen werden können.

Versuchsergebnis:

Ausgehend von dem Ergebnis des vorherigen Versuchs liefert der Median für 10 Messwerte genauere Ergebnisse als der Mittelwert, wobei die Unterschiede für die Positionsbestimmung nicht relevant sind. Aufgrund seiner Eigenschaft ist der Median jedoch robuster gegen Ausreißer, aus diesem Grund ist er dem Mittelwert vorzuziehen.

4.1.3 Position der Beacons

Versuchsbeschreibung:

In diesem Versuch wird untersucht, in welcher Höhe sich die Beacons für ein optimales Ergebnis befinden sollen. Die Position des Endgerätes wird dabei auf etwa 80 - 110 cm Höhe geschätzt, da davon auszugehen ist, dass der Anwender das Gerät bei der Positionsbestimmung in der Hand hält.

Versuchsaufbau:

Das iPhone wird bei diesem Versuch konstant auf einer Höhe von 85 cm platziert.
Die Beacons befinden sich auf einer Höhe von 0 cm, 85 cm, 125 cm, 190 cm und 260 cm. Die Entfernung zum iPhone beträgt 8 Meter.
Die Sendeleistung und das Sendeintervall sind identisch zu den vorherigen Versuchen mit 4 dBm und 100 ms eingestellt. Aufgrund des Ergebnisses aus 4.1.1, 4.1.2 und der ausreichenden zur Verfügung stehenden Zeit für diesen Versuch, wird aus jeweils 20 Messungen der Median ermittelt.
Die folgende Skizze zeigt den Versuchsaufbau:

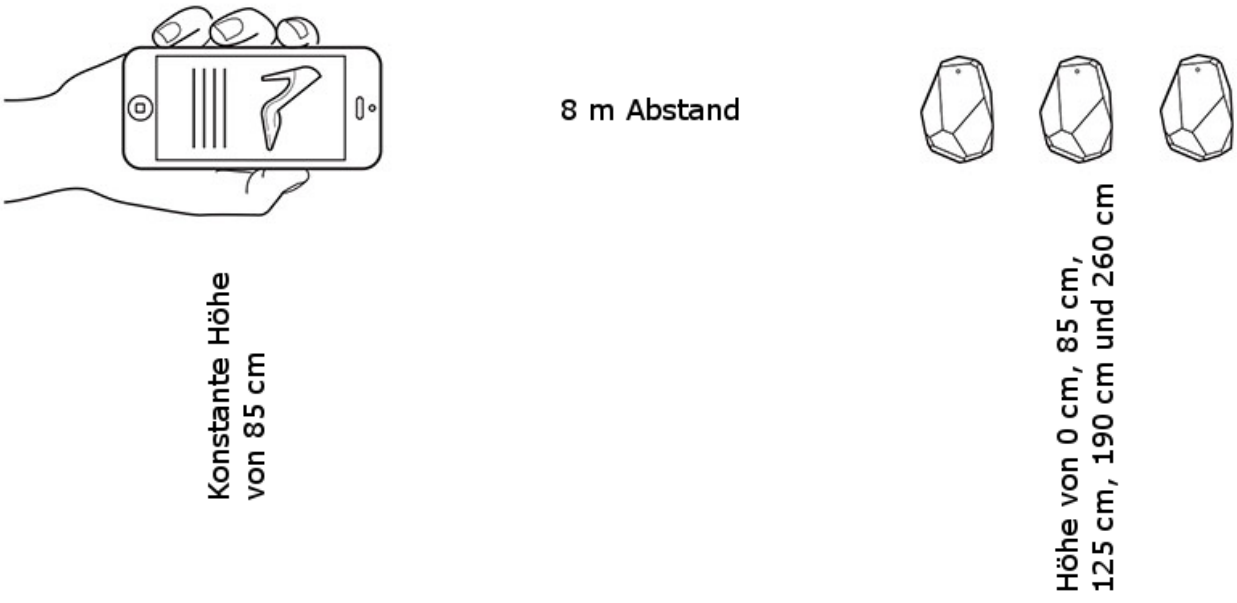


Abbildung 12: Versuchsaufbau des Versuchs 4.1.3
verwendete Grafiken von www.estimote.com

Versuchsdurchführung:

Der Versuch liefert folgende Werte:

4.1.3: (Median der gemessenen Distanzen) - (wirkliche Distanz 8 m)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Höhe Beacon 1-3: 0 cm	17,29	16,68	7,61	13,86
Höhe Beacon 1-3: 85 cm	-5,78	-6,70	-6,03	6,17
Höhe Beacon 1-3: 125 cm	-5,19	-5,87	-4,57	5,21
Höhe Beacon 1-3: 190 cm	3,15	2,27	3,43	2,95
Höhe Beacon 1-3: 200 cm	-2,95	-1,08	-5,21	3,08
Höhe Beacon 1-3: 260 cm	6,88	5,01	3,92	5,27
Mittelwert der Absolutwerte	6,87	6,26	5,12	-

Versuchsergebnis:

In diesem Versuch fällt auf, dass die gemessene Distanz stark von der wirklichen abweicht, wenn die Beacons auf dem Boden liegen. Dies ist daher zu vermeiden. Auffällig ist ebenfalls, dass z.B. mit dem dritten Beacon bei 190 cm eine Differenz von -3,43 m und bei 200 cm eine Differenz von -5,21 m ermittelt wird. Dies bedeutet eine Schwankung um 8,64 m bei einer wirklichen Entfernung der Geräte von 8 Metern. Diese Schwankung unter optimalen Bedingungen wird im fünften Versuch näher untersucht.

Die genausten Ergebnisse werden ermittelt, wenn sich die Beacons auf einer Höhe von 190 cm befinden. Die Höhe könnte zudem von Vorteil sein, da sie über die Durchschnittsgröße eines Menschen reicht. Personen könnten das Signal der Beacons schwächen. Dieses wird im späteren Verlauf der Versuchsreihe ebenfalls untersucht.

4.1.4 Einstellung von Signalstärke und Sendeintervall

Versuchsbeschreibung:

Laut Estimote wird die gemessene Distanz präziser, wenn die Signalstärke erhöht und das Sendeintervall verringert wird [28]. Dies soll in diesem Versuch untersucht werden.

Versuchsaufbau:

Gemessen wird auf den Distanzen 5 und 8 m. Um dem Einfluss durch die Höhe der Beacons zu vermeiden, werden diese, wie das iPhone, konstant auf einer Höhe von 85 cm platziert.

Die Sendeleistung beträgt -12 dBm (Standardeinstellung der Estimote Beacon) und 4 dBm (maximale Sendeleistung der Estimote Beacon).
Das Sendeintervall beträgt 400 ms (Standardeinstellung der Estimote Beacon) und 100 ms (minimalstes einstellbares Sendeintervall der Estimote Beacons).
Dabei ist zu beachten, dass laut Hersteller das minimale Sendeintervall 50 ms beträgt. Bei der Einstellung auf 50 ms springen die Beacons jedoch automatisch auf 100 ms um. Angegeben wird die Differenz des wirklichen Abstands zum Median aus 20 gemessenen Abstandswerten.

Versuchsdurchführung:

4.1.4 (Median der gemessenen Distanzen) - (wirkliche Distanz 5 m)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Signalstärke: 4 dBm Sendeintervall: 100 ms Reichweite: ca. 70 m	-3,99	-3,80	-4,45	4,08
Signalstärke: 4 dBm Sendeintervall: 400 ms Reichweite: ca. 70 m	-4,82	-3,66	-4,90	4,46
Signalstärke: -12 dBm Sendeintervall: 100 ms Reichweite: ca. 15 m	-3,58	-3,50	-4,23	3,77
Signalstärke: -12 dBm Sendeintervall: 400 ms Reichweite: ca. 15 m	-3,86	-3,71	-4,31	3,95
Mittelwert der Absolutwerte	4,06	3,66	4,47	-

4.1.4 (Median der gemessenen Distanzen) - (wirkliche Distanz 8 m)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Signalstärke: 4 dBm Sendeintervall: 100 ms Reichweite: ca. 70 m	-5,75	-5,77	-6,85	6,12
Signalstärke: 4 dBm Sendeintervall: 400 ms Reichweite: ca. 70 m	-7,66	-5,08	-6,73	6,48
Signalstärke: -12 dBm Sendeintervall: 100 ms Reichweite: ca. 15 m	-5,45	-5,36	-6,69	5,83
Signalstärke: -12 dBm Sendeintervall: 400 ms Reichweite: ca. 15 m	-5,89	-5,64	-6,93	6,15
Mittelwert der Absolutwerte	6,18	5,46	6,8	-

Versuchsergebnis:

Die gemessene Distanz ist bei einem Sendeintervall von 100 ms tatsächlich genauer, als bei einem Sendeintervall von 400 ms. Allerdings liegen die Schwankungen unter 0,5 m, sodass zugunsten des Stromverbrauchs nicht immer das höchste Sendeintervall gewählt werden muss. Hier sollte auf den Anwendungsfall Rücksicht genommen werden. Wird das Smartphone etwa bei der Positionsbestimmung bewegt, empfiehlt sich, ein höheres Sendeintervall zu wählen als bei einem festpositionierten Smartphone.

Ein Wert von -12 dBm liefert zwar einen genaueren Wert als ein Wert von 4 dBm, allerdings sind die Unterschiede sehr gering.
Aufgrund des höheren Stromverbrauches bei einer hohen Sendeleistung und der nicht relevanten Unterschiede von etwa 20 cm, sollte auch hier auf den Anwendungsfall geachtet werden. Entscheidend hierfür ist die Frage, wie weit das Signal der Beacons gesendet werden soll.

4.1.5 Präzision der Messungen unter optimalen Bedingungen

Versuchsbeschreibung:

In diesem Versuch wird untersucht, ob eine Abstandsmessung mit Hilfe der Estimote Beacons unter optimalen Bedingungen präzise genug für die Positionsbestimmung über das Verfahren der Lateration ist.

Versuchsaufbau:

Um ein realistisches Ergebnis für die Positionsbestimmung zu erhalten, werden die gleichen Rahmenbedingungen, wie für die App vorgesehen, verwendet. Der Median wird aufgrund der schnelleren Verarbeitungszeit aus zehn Werten ermittelt. Das iPhone wird auf etwa 90 cm Höhe in der Hand gehalten. Die Beacons befinden sich auf einer Höhe von 195 cm. Als Signalstärke wird 4 dBm und als Sendeintervall 100 ms verwendet. Der Abstand wird für die Entfernung zwischen iPhone und Beacons für 10 m, 20 m, 30 m, 40 m, 50 m und 58 m ermittelt.

Versuchsdurchführung:

4.1.5: 1. Messung (Median der gemessenen Distanzen) - (wirkliche Distanz)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Distanz: 10 m	2,10	-0,43	0,45	0,99
Distanz: 20 m	6,06	6,00	4,68	5,57
Distanz: 30 m	14,91	10,56	-1,10	8,85
Distanz: 40 m	6,73	5,16	-3,72	5,2
Distanz: 50 m	-2,21	-13,32	-15,42	10,31
Distanz: 58 m	-17,60	-21,98	-28,02	22,53
Mittelwert der Absolutwerte	8,26	9,57	8,89	-

4.1.5: 2. Messung (Median der gemessenen Distanzen) - (wirkliche Distanz)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Distanz: 10 m	10,99	11,07	6,83	9,63
Distanz: 20 m	-4,57	-4,78	-6,80	5,38
Distanz: 30 m	12,35	4,94	-4,90	7,39
Distanz: 40 m	20,45	0,84	3,69	8,32
Distanz: 50 m	-15,79	-16,06	-14,07	15,3
Distanz: 58 m	-12,08	-9,80	-17,63	13,17
Mittelwert der Absolutwerte	12,7	7,91	8,98	-

4.1.5: 3. Messung (Median der gemessenen Distanzen) - (wirkliche Distanz)

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Distanz: 10 m	-0,87	0,29	-3,51	1,55
Distanz: 20 m	-7,44	-5,33	-8,42	7,06
Distanz: 30 m	15,64	34,23	25,97	25,28
Distanz: 40 m	23,32	6,86	10,48	13,55
Distanz: 50 m	3,79	-7,36	-1,69	4,28
Distanz: 58 m	-5,25	-7,51	-3,59	5,45
Mittelwert der Absolutwerte	9,38	10,26	8,94	-

Versuchsergebnis:

Auffällig ist, dass die Präzision der gemessenen Distanz sehr unterschiedlich ausfällt. Dies betrifft vor allem die Messungen bei einer Distanz über 30 Metern. Auch sind zwischen den einzelnen Beacons, trotz denselben Bedingungen, sehr starke Schwankungen festzustellen. Dies deutet darauf hin, dass der Algorithmus der Estimote SDK, der zur Abstandsmessung verwendet wird, sehr ungenau ist oder die RSSI der Beacons, mit welcher der Abstand berechnet wird, sehr starken Schwankungen unterlegen ist.

4.1.6 Fazit: Grundlegende Versuche

Als Fazit dieser Versuchsreihe lässt sich festhalten, dass zur Abstandsmessung nicht nur ein Wert betrachtet werden sollte, sondern mehrere, da es vereinzelt zu Ausreißern kommt. Dabei reicht es aus, wenn aus zehn Messwerten der Median ausgewertet wird. Die Estimote Beacons sollten dabei wie von Estimote vorgeschlagen angebracht werden. Die optimale Höhe beträgt etwa 195 cm.
Es ist jedoch aufgefallen, dass die Abstandsmessung mit Hilfe der Estimote Beacons sehr unterschiedlich ausfällt. Zum Teil sind sehr starke Schwankungen der gemessenen Distanz vorhanden. Gerade bei Distanzen von über 30 Metern ist die Abstandsmessung selbst ohne Störquellen starken Schwankungen unterlegen.
In weiteren Versuchen ist zu klären, ob diese Schwankungen es ermöglichen, das Verfahren der Lateration anzuwenden. Des Weiteren muss beobachtet werden, wie viele Messungen das iPhone für eine möglichst genaue Berechnung der Distanz benötigt, wenn es in Bewegung ist. Auch ist zu klären, wie sich die gemessenen Abstände unter dem Einfluss von Störquellen verändern.

Für kleinere Distanzen vom iPhone zu einem Beacon ist das Verfahren der Lateration nicht zu empfehlen. Davon ausgehend, dass sich die Beacons nicht in der Nähe befinden, würden die zwei weiter entfernten Beacons die Position verfälschen. Hier empfiehlt es sich, als gemessene Position die Position des am nächsten liegenden Beacons zu verwenden.

4.2 Einfluss von Signaldämpfern

In dieser Versuchsreihe wird untersucht wie Dämpfungen den gemessen Abstand zwischen iPhone und Estimote Beacons beeinflussen. Als Signaldämpfer werden Wände und Personen verwendet.

4.2.1 Einfluss von Wänden

Versuchsbeschreibung:

In diesem Versuch wird untersucht inwieweit Wände den gemessenen Abstand vom iPhone zu den Estimote Beacons verändern.
Es ist davon auszugehen, dass der gemessene Abstand größer wird.

Versuchsaufbau:

Gemessen wird die Distanz von 9 Metern. Nach 2,4 m befindet sich eine Glastür in einer 20 cm dicken Steinwand und nach 6,50 m eine Holztür in einer 15 cm dicken Steinwand. Nach 8 m befindet sich erneut eine 15 cm dicke Steinwand. Es werden Messungen mit geöffneten und Messungen mit geschlossenen Türen durchgeführt.
Die Signalstärke der Beacons beträgt 4 dBm und das Sendeintervall 100 ms. Sie befinden sich auf einer Höhe von 200 cm und wurden wie von Estimote vorgeschlagen angebracht. Das iPhone wird auf einer Höhe von 90 cm in der Hand gehalten.

Versuchsdurchführung

4.1.6: Wände als Störquellen - Median

	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
Türen geöffnet	39,89	56,73	66,26	54,29
Glastür geschlossen Holztür geöffnet	-	-	-	-
Holztür geschlossen Glastür geöffnet	-	-	-	-

Versuchsergebnis:

Wie erwartet fallen die gemessenen Distanzen sehr groß aus. Als Vergleich können die Messungen aus dem Versuch 4.1.5 auf 10 m genommen werden, die trotz einer höheren wirklichen Distanz deutlich genauer ausfallen.
Sobald eine Tür verschlossen wird, befinden sich die Beacons nicht mehr in der Reichweite des iPhones.

4.2.2 Einfluss von Personen

Versuchsbeschreibung & Versuchsaufbau:

In dieser Versuchsreihe soll untersucht werden, im welchem Ausmaß Personen zwischen den Estimote Beacons und dem iPhone den gemessenen Abstand zwischen den Geräten verändern. Dazu wurden zwei Versuchsorte ausgewählt. Zum einem ein Foyer mit einer Länge von 32 m und einer Glastür bei 16,5 m. Ausgemessen werden dabei die Abstände 16 m und 32 m. Zum anderem ein belebter Ort, der eine Distanzmessung von 18,8 m Länge erlaubt.
Die Signalstärke beträgt bei beiden Versuchen 4 dBm und das Sendeintervall 100 ms. Die Beacons sind an einer Wand auf 200 cm Höhe angebracht. Die Ausrichtung erfolgt, wie von Estimote vorgeschlagen. Das iPhone wird während der Messungen in der Hand auf etwa 90 cm Höhe gehalten.
Es ist bei beiden Versuchsorten zu erwarten, dass der gemessene Abstand größer wird, wenn sich Personen zwischen dem iPhone und den Beacons befinden. Ausgewertet wird jeweils die Differenz des Median aus 20 gemessenen Werten zum wirklichem Abstand.

4.2.2.1 Einfluss von Personen - 1. Versuchsort

Versuchsaufbau:

Es werden die Distanzen 16 m und 32 m gemessen. Einmal mit Personen und einmal als Kontrollmessung ohne Personen. Die Personen sind dabei in der Breite auf etwa 4 m verteilt. Bei der Messung über die Distanz von 32 m befindet sich auf 16,5 m eine geöffnete Glastür.

Versuchsdurchführung:

4.2.2.1: Personen als Störquellen - Distanz 16 m und 32 m - Median

Nr.	Beschreibung	Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
1	**16 m ohne Personen**	-3,93	-6,89	-8,48	6,43
2	**16 m ca. 20 Personen 1. Messung**	-7,50	-8,71	-3,39	6,53
3	**16 m ca. 50 Personen 2. Messung**	-0,49	-2,55	-4,72	2,58
4	**32 m ohne Personen**	1,4	-2,34	-4,19	2,64
5	**32 m ca. 50 Personen 1. Messung**	20,73	10,40	20,20	17,11
6	**32 m ca. 50 Personen 2. Messung**	9,34	25,59	23,87	19,59
7	**32 m ca. 15 Personen 1 Person direkt vor dem iPhone 3. Messung**	24,26	29,69	28,85	27,6
	Mittelwert der Absolutwerte	9,66	12,31	13,38	-

Versuchsergebnis:

Wie zu erwarten wird der gemessene Abstand größer, wenn sich ca. 50 Personen zwischen dem iPhone und den Beacons befinden. Bei einer Distanz von 16 m wird dabei sogar die Abweichung zum wirklichen Abstand der Geräte geringer. Dies hängt damit zusammen, dass ohne Personen der Abstand deutlich zu gering gemessen wurde. Der Aufenthalt von 20 Personen zwischen dem iPhone und den Estimote Beacons hat jedoch keinen Einfluss auf die gemessene Distanz.

Bei einer Distanz von 32 m zwischen iPhone und Estimote Beacons wird der gemessene Abstand ebenfalls größer, wenn sich Personen zwischen den Geräten befinden. Aufgrund dessen, dass die Abweichung von der gemessenen Distanz zur wirklichen Distanz ohne Personen mit ca. 2,64 m sehr gering ist, wird der Abstand nun deutlich zu groß gemessen, sofern sich Personen zwischen den Geräten aufhalten.
Auffällig ist das Ergebnis des Versuchs Nr. 7. Trotz einer geringeren Personenanzahl ist die Differenz des gemessenen Abstandes zum wirklichen Abstand deutlich größer. Dies ist darauf zurückzuführen, dass sich im Versuch Nr. 7 eine Person direkt vor dem iPhone befindet, und

dass sich das iPhone in niedriger Höhe befindet. Bei den Messungen 1-6 ist dies nicht der Fall.
Befinden sich Personen direkt vor den Beacons hat dies kaum Auswirkungen auf die Messung, da die Beacons mit 200 cm höher positioniert sind, als die Personen groß sind. Das Signal der Beacons geht demnach über sie hinweg.

4.2.2.2 Einfluss von Personen - 2. Versuchsort

Versuchsaufbau:

Die Beacons und das iPhone befinden sich in einem Abstand von 18,8 m zueinander. Insgesamt werden vier Messungen durchgeführt. Zwei Messungen ohne Personen zwischen den Geräten und zwei Messungen mit etwa 70 Personen, die sich zwischen den Geräten befinden. Die Personen stehen dabei nicht nur in einer Linie zwischen dem iPhone und den Beacons, sondern verteilen sich ebenfalls seitlich.

Versuchsdurchführung:

4.2.2.2: Personen als Störquellen - Distanz 50 m und 72 m - Median

Nr		Abweichung Beacon 1	Abweichung Beacon 2	Abweichung Beacon 3	Mittelwert der Absolutwerte
1	**18,8 m ohne Personen 1. Messung**	-0,34	-6,81	-4,27	3,8
2	**18,8 m ohne Personen 2. Messung**	3,17	0,87	0,94	1,66
3	**18,8 m ca. 70 Personen 1. Messung**	14,11	6,62	5,34	8,69
4	**18,8 m ca. 70 Personen 2. Messung**	7,27	5,41	3,65	5,44
	Mittelwert der Absolutwerte	6,22	4,92	3,55	-

Versuchsergebnis:

In diesem Versuch bestätigt sich, wie auch im Versuch 4.2.2.1, dass die Personen das Signal der Beacons dämpfen und dadurch ein größerer Abstand errechnet wird.
Auffällig ist, dass bei der dritten Messung die Beacons 1 und 2 bzw. 3 sehr unterschiedliche Differenzen gemessen haben. Dies bestätigt noch einmal die starken Schwankungen, welche bereits im Versuch 4.1.5 auftraten.

4.2.3 Fazit: Einfluss von Signaldämpfern

Als Fazit dieser Versuchsreihe lässt sich ziehen, dass Wände das Signal der Beacons sehr stark dämpfen. Aus diesem Grund ist es zu empfehlen, dass in jeden Raum Beacons installiert werden und bei der Positionsbestimmung als erster Schritt der Raum bestimmt wird.

Des Weiteren wird festgestellt, dass das Signal der Beacons durch Personen gedämpft wird und dadurch ein großer Abstand zwischen iPhone und Beacons ermittelt wird. So zeigen z.B. die Werte im Versuch 4.2.2.1 mit Personen eine deutlich größere Distanz auf, als ohne Personen. Die Messergebnisse aus dem Versuch 4.1.5 zeigen jedoch auch, dass solche Schwankungen auch ohne Hindernisse auftreten können.
Besonders auffällig ist jedoch im Versuch 4.2.2.1, dass das Signal der Beacons stark gedämpft wird, wenn sich eine oder mehrere Personen direkt vor dem iPhone befinden.

4.3 Versuche mit iBeacon zur Positionsbestimmung

Versuchsbeschreibung:

In dieser Versuchsreihe soll untersucht werden, ob die Estimote Beacons zur Positionsbestimmung in Gebäuden geeignet sind. Dazu werden in zwei Versuchen zwei Verfahren zur Positionsbestimmung durchgeführt. Zum einen das Verfahren der Lateration. Zum anderen das Fingerprinting-Verfahren.

Versuchsaufbau:

Die beiden Versuche haben den gleichen grundlegenden Versuchsaufbau. So werden beide Versuche in einem 8,4 x 18 m großem Raum durchgeführt.
Die drei Estimote Beacons sind dabei auf einer Höhe von 200 cm und mit dem Punkt nach oben an den Wänden angebracht (Abbildung 10). Sie haben folgende Positionen:

Beacon 1: X = 0 m; Y = 5,8 m
Beacon 2: X = 6,4 m; Y = 0 m
Beacon 3: X = 4,6 m; Y = 17,6 m

Das Sendeintervall der Beacons beträgt 100 ms. Die Sendeleistung beträgt 4 dBm. Für diese Versuchsreihe wird ein iPad 4 verwendet.

4.3.1 Positionsbestimmung durch das Verfahren der Lateration

Versuchsbeschreibung:

Im diesen Versuch soll untersucht werden, ob sich das Verfahren der Lateration aus Kapitel 2.4.1 zur Positionsbestimmung in Gebäuden eignet. Im Wesentlichen soll festgestellt werden, ob die Radien der drei Beacons Schnittpunkte bilden, die auf eine mögliche Position des Endgeräts schließen lassen. Bei diesem Versuch werden die Position des Endgerätes, die Distanzen zu den Beacons, die Schnittpunkte der Beacons und die errechnete Position des iPads dokumentiert.

Versuchsaufbau:

Es gilt der Versuchsaufbau aus Kapitel 4.3. Die Positionsbestimmung wird dabei von verschiedenen Punkten durchgeführt. Die Distanz vom iPad zum Beacon wird dabei aus dem Median von zehn gemessenen Distanzwerten ermittelt.

Versuchsdurchführung:

4.3.1 Mediane aus zehn gemessenen Distanzen

Position des iPads	Beacon 1	Beacon 2	Beacon 3
(2,6 l 2,6)	1,33	1,59	2,81
(3,6 l 13,6)	6,50	2,41	0,93
(4,6 l 9,6)	1,24	3,45	1,46
(4,6 l 16,6)	3,43	3,23	0,30
(5,6 l 5,6)	0,67	2,64	2,72
(6,6 l 6,6)	0,77	3,15	2,48
(6,6 l 10,6)	2,98	3,31	0,69

4.3.1: Position des iPads (x l y) / Schnittpunkte (x l y) zwischen den jeweilgien Beacons

Position des iPads	1. Punkt: Beacon 1 - 2	2. Punkt: Beacon 1 - 2	1. Punkt: Beacon 1 - 3	2. Punkt: Beacon 1 - 3	1. Punkt: Beacon 2 - 3	2. Punkt: Beacon 2 - 3
(2,6 l 2,6)	-	-	-	-	-	-
(3,6 l 13,6)	(5,40 l 2,20)	(4,12 l 0,78)	-	-	-	-
(4,6 l 9,6)	-	-	-	-	-	-
(4,6 l 16,6)	-	-	-	-	-	-
(5,6 l 5,6)	-	-	-	-	-	-
(6,6 l 6,6)	-	-	-	-	-	-
(6,6 l 10,6)	-	-	-	-	-	-

4.3.1: Ergebnis der Positionsbestimmung mittels Lateration

Position des iPads	Errechnete Position des iPads	Euklidische Distanz
(2,6 l 2,6)	-	-
(3,6 l 13,6)	-	-
(4,6 l 9,6)	-	-
(4,6 l 16,6)	-	-
(5,6 l 5,6)	-	-
(6,6 l 6,6)	-	-
(6,6 l 10,6)	-	-

Versuchsergebnis:

Wie aus den Ergebnissen der vorherigen Versuche zu erwarten, weicht die gemessene Distanz der Beacons zum iPad zur wirklichen Distanz so stark ab, dass das Verfahren der Lateration nicht möglich ist.
Es wäre jedoch denkbar, dass Verfahren abzuwandeln und zu optimieren. So könnten die Distanzen der Beacons erweitert werden, wenn die Kreise keine Schnittpunkte bilden und sie nicht ineinander liegen. Auch könnten kürzere Distanzen höher gewichtet werden.
So ist das Beacon 3 beim Punkt (4,6 l 16,6) sehr nah am iPad. Da die Position des Beacons 3 bekannt ist (4,6 l 17,6), lässt sich so auf die Position des iPads schließen. Allerdings hat letzteres Vorgehen nichts mehr mit dem Verfahren der Lateration gemeinsam.

4.3.2 Positionsbestimmung durch das Fingerprinting-Verfahren

Versuchsbeschreibung:

In diesem Versuch wird untersucht, ob durch das Fingerprinting-Verfahren, welches in Kapitel 2.4.2 beschrieben wird, eine effektive Positionsbestimmung mit Hilfe der iBeacon-Technologie möglich ist.
Hierfür wird der Versuch in zwei Teile aufgeteilt. Zuerst wird die Erstellung der Radio Map durchgeführt, welche eine Voraussetzung für den zweiten Teil des Versuchs ist.
Im zweitem Teil des Versuchs wird überprüft, ob mit Hilfe der Radio Map die Position des iPads innerhalb des Raumes bestimmt werden kann.

Die beiden Versuchsteile wurden an aufeinander folgenden Tagen durchgeführt. Dadurch kann angenommen werden, dass die Radio Map wiederverwendet werden kann und nicht vor jeder Positionsbestimmung neu ermittelt werden muss.

4.3.2.1 Erstellen der Radio Map

Versuchsbeschreibung:

In diesem Versuch wird mit Hilfe des iPads eine Radio Map erstellt. Dies ist erforderlich, um eine Positionsbestimmung durch das Fingerprinting-Verfahren durchführen zu können.

Versuchsaufbau:

Es gilt der Versuchsaufbau aus Kapitel 4.3. In dem vorhanden Raum wird ein Koordinatensystem mit der Größe X= 7,2 m und Y= 16,4 m gelegt. Innerhalb dieses Koordinatensystems befinden sich die Anchor points. Das Koordinatensystem beginnt im Raum bei X= 0,6 m und Y= 0,6 m und endet somit bei X= 7,8 m und Y= 17 m. Die Randbereiche des Raumes werden somit nicht vermessen. Dies ist nicht notwendig, da diese Punkte später bei Bedarf durch Interpolation errechnet werden können.
Die Beacons befinden sich, wie in Kapitel 4.3 beschrieben, weiterhin an der Wand und somit außerhalb des Koordinatensystems.
Innerhalb dieses Koordinatensystems werden 36 Anchor Points verteilt. Der Abstand eines Anchor points zum nächsten beträgt jeweils 2 m. So ergeben sich folgende Positionen der Anchor points:

1. X= 0 m; Y= 0 m
2. X= 0 m; Y= 2 m
3. X= 0 m; Y= 4 m

…

10. X= 2 m; Y= 0 m
11. X= 2 m; Y= 2 m

…

36. X= 6 m; Y= 16 m

Versuchsdurchführung:

4.3.2.1: Median aus zehn gemessenen Abständen der Beacons zum Anchor

Nr.	Anchor Point (x,y)	Beacon 1	Beacon 2	Beacon 3
1	0,0	4,27	3,77	6,60
2	0,2	1,09	2,28	1,62
3	0,4	0,71	1,93	1,65
4	0,6	0,71	2,21	2,18
5	0,8	3,26	4,14	2,02
6	0,10	3,70	5,80	2,00
7	0,12	7,07	4,99	0,69
8	0,14	10,20	4,46	0,57
9	0,16	10,93	4,75	0,54
10	2,0	1,08	1,43	1,52
11	2,2	1,01	1,00	5,01
12	2,4	0,92	2,69	3,02
13	2,6	0,69	1,86	2,28
14	2,8	1,82	2,18	1,81
15	2,10	3,95	6,76	0,91
16	2,12	6,06	3,51	0,68
17	2,14	6,46	4,25	0,35
18	2,16	8,24	4,35	0,60
19	4,0	2,27	0,57	2,28
20	4,2	0,54	1,35	1,68
21	4,4	0,65	1,79	1,57
22	4,6	1,05	2,10	2,59
23	4,8	0,84	2,40	1,37
24	4,10	1,27	4,39	0,88
25	4,12	2,49	3,40	0,67
26	4,14	4,12	3,80	0,38
27	4,16	5,87	4,91	0,22
28	6,0	1,40	0,86	4,49
29	6,2	1,12	1,44	3,62

Nr.	Anchor Point (x,y)	Beacon 1	Beacon 2	Beacon 3
30	6,4	0,48	1,78	3,79
31	6,6	0,79	2,32	3,37
32	6,8	0,79	2,01	1,63
33	6,10	1,66	3,50	2,75
34	6,12	2,26	4,17	0,90
35	6,14	2,93	5,58	0,46
36	6,16	5,34	6,12	0,50

Versuchsergebnis:

Auffällig ist auch in diesem Versuch, dass die gemessenen Distanzen stark von den wirklichen Distanzen abweichen. Dies wird jedoch nicht weiter beachtet, da es für das Fingerprinting-Verfahren keine Bedeutung hat.

Ausgehend von den Positionen der Beacons und der Position des Anchor points sind unerwartete Werte in der Tabelle blau unterlegt.

So sind bei den Positionen 1 und 2 die Abstände zum Beacon 2 auffällig, weil sich die Positionen näher am Beacon 2 befinden, als Position 3 und dennoch ein größerer Abstand ermittelt wurde. Zu erwarten wären Werte unter 1,80 m. Gleiches konnte für die 24. und die 31. Position und das Beacon 2 festgestellt werden.

Auch für die Positionen 21 und 22 fallen die Abstände des Beacon 1 im Verhältnis zu groß aus.

Zu klein fallen hingegen die Abstände bei den Positionen 2, 3 und 10 für das Beacon 3 aus.

4.3.2.2 Positionsbestimmung mit Hilfe der Radio Map

Versuchsbeschreibung:

In diesem Versuch wird untersucht, ob mit Hilfe der Radio Map aus dem Versuch 4.3.2.1 auf die Position des iPads geschlossen werden kann.

Versuchsaufbau:

Es wird der Versuchsaufbau aus dem Kapitel 4.3 verwendet. Untersucht werden die gleichen Positionen wie im Versuch 4.3.1.

Versuchsdurchführung:

4.3.2.2: Ergebniss der Positionsbestimmung mittels Fingerprinting

Nr	Position des iPads	Errechnete Position des iPads	Euklidische Distanz
1	**(2,6 l 2,6)**	(6,6 l 4,6)	4,47
2	**(3,6 l 13,6)**	(2,6 l 16,6)	3,16
3	**(4,6 l 9,6)**	(6,6 l 10,6)	2,24
4	**(4,6 l 16,6)**	(0,6 l 14,6)	4,47
5	**(5,6 l 5,6)**	(0,6 l 6,6)	5,10
7	**(6,6 l 6,6)**	(2,6 l 6,6)	4,00
8	**(6,6 l 10,6)**	(4,6 l 12,6)	2,83

Versuchsergebnis:

Das Fingerprinting-Verfahren liefert relativ genaue Ergebnisse. Die maximale Abweichung beträgt 5,10 m und die minimale beträgt 2,24 m. Vor allem ist zu beachten, dass für die Positionen 2, 3 und 5 keine Messungen in der Radio Map vorliegen und auch keine Interpolation durchgeführt wurde. Das heißt, der nächste Wert, der für die Positionen 2 und 5 angegeben wird, muss mindestens um 1,41 m abweichen. Für die Position 3 gilt dasselbe um 1 m.

4.4 Fazit

Die Versuchsreihen zeigen auf, dass sich das reine Verfahren der Lateration nicht für die iBeacon-Technologie unter Verwendung der Estimote Beacons zur Positionsbestimmung eignet. In der Versuchsreihe konnte kein einziger Punkt ermittelt werden.
Das Fingerprinting-Verfahren liefert dagegen immer eine Position. Dies hängt damit zusammen, dass immer der ähnlichste Wert aus der Radio Map gewählt wird.
Dieses Ergebnis ist für die Positionsbestimmung in der Regel ausreichend. Die ermittelten Positionen weichen jedoch im Einzelnen sehr stark von der wirklichen Position ab. Dies hängt zum einem mit den Schwankungen der gemessenen Abstände zwischen iPad und Beacons zusammen. Zum anderen würden mehrere Anchor Points oder eine Interpolation das Verfahren verfeinern. Auch wäre es denkbar das Fingerprinting-Verfahren zu optimieren, indem die Positionen mit Ausreißern in der Radio Map erneut gemessen werden.

5 Konzeption Prototyp

5.1 Aufgabe & Anforderungsanalyse

5.1.1 Ist-Analyse

Bei Großveranstaltungen ist aufgrund der erhöhten Menschenansammlung mit mehr Patienten zu rechnen, als für diesen Ort und Zeitpunkt ohne Veranstaltung üblich. Aus diesem Grund muss unter gewissen Umständen ein Sanitätsdienst durchgeführt werden, dessen primäres Ziel es ist, den Regelrettungsdienst der Stadt oder des Kreises zu entlasten. Die zusätzlich eingesetzten Sanitäter sind zudem in der Lage schneller Hilfe zu leisten als der Regelrettungsdienst, da sie sich direkt am Ort der Großveranstaltung befinden. Dies gilt als weiterer Vorteil, da es in einer Notfallsituation wichtig ist, dass der Patient unverzüglich versorgt wird. Die zivilen Ersthelfer spielen dabei in den ersten Minuten des Notfalls eine entscheidende Rolle. So haben sie unter anderem die Aufgabe, medizinisches Fachpersonal anzufordern. Dies wären im optimalen Fall die Sanitäter des Sanitätsdienstes, welche sich in der Regel bei einer Sanitätsstation befinden. Das Auffinden der nächsten Sanitätsstation oder auch des nächsten Notausganges ist für den zivilen Ersthelfer im Notfall jedoch sehr umständlich und zeitaufwendig, da dieser sich in der Regel auf dem Gelände kaum auskennt und sich vorher nicht über Sanitätsstationen oder Notausgänge informiert hat. Die große Ansammlung von Personen zählt als zusätzlicher Faktor, der das Erreichen des Zielorts gerade in unbekannten Gebäuden erschwert. Kann der Ersthelfer keinen Sanitäter auffinden, hat er die Alternative, den Notfall über die Notruftelefonnummer 112 zu melden. Dabei kann der Notruf problemlos von der Leitstelle an die Sanitäter weitergeleitet werden. Allerdings ist die mündliche Beschreibung des Notfallorts durch den Ersthelfer dabei häufig sehr aufwendig und ungenau. Das Problem der Navigation durch die große Ansammlung von Menschen wird damit an die Sanitäter weitergeleitet, indem diese nun die Aufgabe haben, mit ungenauen Ortsangaben den Patienten zu lokalisieren.

5.1.2 Soll-Analyse

Der im Rahmen dieser Arbeit entstehende Prototyp soll durch den Nutzen von iBeacon auf einer Veranstaltung mit Sanitätsdienst das Alarmieren der Sanitäter durch den Ersthelfer im Falle eines Notfalls erleichtern. In besonderem Maße soll die Ermittlung des Notfallorts präzisiert und für den Ersthelfer vereinfacht werden. Dadurch sollen die Ziele erreicht werden, den Notruf schneller an die Sanitäter zu übermitteln und durch eine präzise Angabe des Notfallorts das Auffinden des Patienten für die Sanitäter zu erleichtern. Dies geschieht insbesondere durch eine Navigation, die den Sanitätern die Richtung des Patienten weist. Über eine Notrufliste erhalten die Sanitäter eine Übersicht über alle Alarmierungen mit den jeweiligen Patienten. Der Ersthelfer soll dabei aus Datenschutzgründen keine Einsicht in die Notrufinformationen anderer Ersthelfer erhalten. Um diese Anforderungen zu erfüllen, sind zwei Apps auf verschiedenen Endgeräten erforderlich.
Zum einen eine App für die Sanitäter, die sie über neue Notrufe informiert, sie zum Patienten navigiert und ihnen eine Einsicht in die Patientenliste realisiert. Zum anderen ist eine App für die Ersthelfer erforderlich, die es diesen ermöglicht, die fünf Ws des Notrufs an die Sanitäter zu senden. Diese allgemein gültigen Ws sind eine Merkhilfe für den Ersthelfer und sollen der Leitstelle einen Überblick über den Notfall geben, um das passende Material und Personal bereitstellen zu können. Sie bestehen aus folgenden Punkten:

1) Wo ist der Notfallort?
2) Was ist geschehen?
3) Wie viele Verletzte oder Erkrankte?
4) Welche Art von Verletzungen oder Erkrankungen?
5) Warten auf Rückfragen! [29]

Zusätzlich bietet die Helfer-App aus Sicherheitsgründen die Funktion an, die zuständige Leitstelle über die 112 telefonisch zu erreichen. Des Weiteren ist die Helfer-App in der Lage, die Richtung zum nächsten Notausgang mittels eines Pfeils anzuzeigen.
Für die App der Sanitäter empfiehlt es sich, ein großes Display zu wählen, um die Patientenliste übersichtlich zu gestalten und gegebenenfalls in einer späteren Version des Prototyps einen Lageplan einzubauen. Für die Helfer-App ist kein großes Display erforderlich, da keine Patientenliste und kein Lageplan angezeigt werden. Es empfiehlt sich dagegen ein Endgerät mit einer handlichen Displaygröße zu wählen, welches der Ersthelfer immer bei sich tragen kann. Zusätzlich ist erforderlich, dass der Ersthelfer mit dem

Endgerät eine Telefonverbindung mit der Leitstelle über die 112 aufbauen kann.

5.2 Beschreibung der Helfer-App

Die Helfer-App hat die Anwendungsfälle Alarmiern der Sanitäter, Absetzen eines Notrufs und Navigation zum Notausgang. Im Folgenden werden diese Anwendungsfälle näher beschrieben.

5.2.1 Use Case: Alarmierung der Sanitäter

Eine der wichtigsten Funktionen der Helfer-App ist es, den Notruf, im Rahmen der fünf Ws, an die Sanitäter zu übermitteln, sodass diese sich möglichst schnell am Einsatzort einfinden können.

Vorbedingungen:

Als Voraussetzung für diesen Anwendungsfall muss der Ersthelfer ein iBeacon kompatibles Endgerät mit einer aktiven Internetverbindung, sowie mit eingeschaltetem Bluetooth besitzen. Zur erfolgreichen Positionsbestimmung ist es erforderlich, dass sich der Ersthelfer mit seinem Endgerät in der Reichweite von mindestens einem iBeacon befindet. Dieser Anwendungsfall kann über die Funktion Notruf gestartet werden.

Use Case:

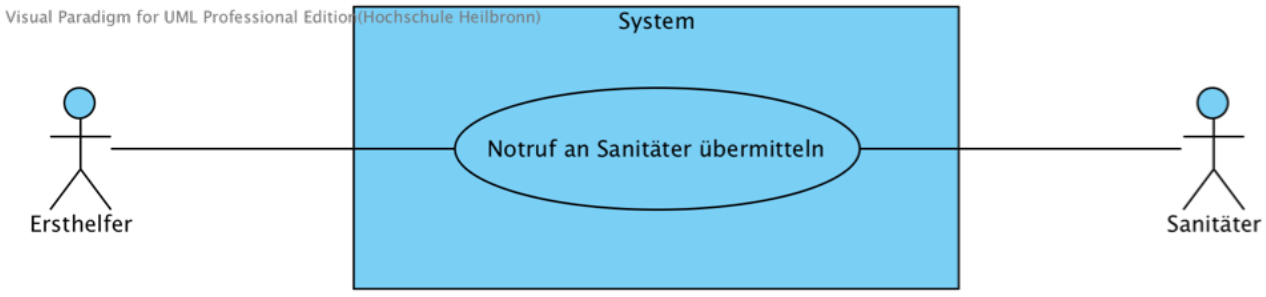


Abbildung 13: Use Case-Diagramm Sanitäter alarmieren

Beschreibung:

In dem Anwendungsfall *Alarmieren der Sanitäter* werden die fünf Ws an die Sanitäter übermittelt.
Die Beantwortung nach dem Notfallort erfolgt dabei automatisch durch die Helfer-App. Hierzu wird die Position des Endgerätes vom Ersthelfer über die in der Nähe befindlichen iBeacons ermittelt.
Diese Positionsbestimmung erfolgt im Hintergrund, sodass der Ersthelfer als ersten Schritt die Patientenzahl bestimmt. Er kann dabei zwischen einem, zwei, drei und vier oder mehr Patienten wählen. Wählt der Ersthelfer drei oder mehr als vier Patienten aus, werden zu den Punkten 2) und 4) keine Fragen gestellt. Die individuelle Beantwortung für alle Patienten wäre für den Ersthelfer zu zeitaufwendig. Stattdessen gibt der Ersthelfer seinen Namen und seine Telefonnummer für Rückfragen an. Die Sanitäter können so telefonisch weitere Informationen erfragen oder direkt beim Notfallort eine Triage[7] durchführen.
Sind dagegen nur ein bis zwei Patienten am Notfallort, kann der Ersthelfer durch eine Auswahl von Notfallsituationen, Erkrankungen und Verletzungen den Notfall näher beschreiben. Die Beschreibung erfolgt dabei individuell für jeden Patienten. Ebenfalls gibt der Ersthelfer zum Schluss seinen Namen und seine Telefonnummer für weitere Rückfragen an.

Die App übermittelt die Informationen des Notrufes an den Webserver.

Nachbedingungen:

Der Notruf wurde zum Webserver übermittelt.
Für nachfolgende Notrufe werden die Telefonnummer und der Name des Ersthelfers gespeichert. Bei einem erneuten Notruf werden die Informationen direkt vorgeschlagen, können jedoch verändert und überschrieben werden.

[7] Triage: Sichtung von Patienten

5.2.2 Aktivitätsdiagramm: Alarmieren der Sanitäter

Das folgende Diagramm beschreibt das *Alarmieren der Sanitäter* in Form eines Aktivitätsdiagramms.

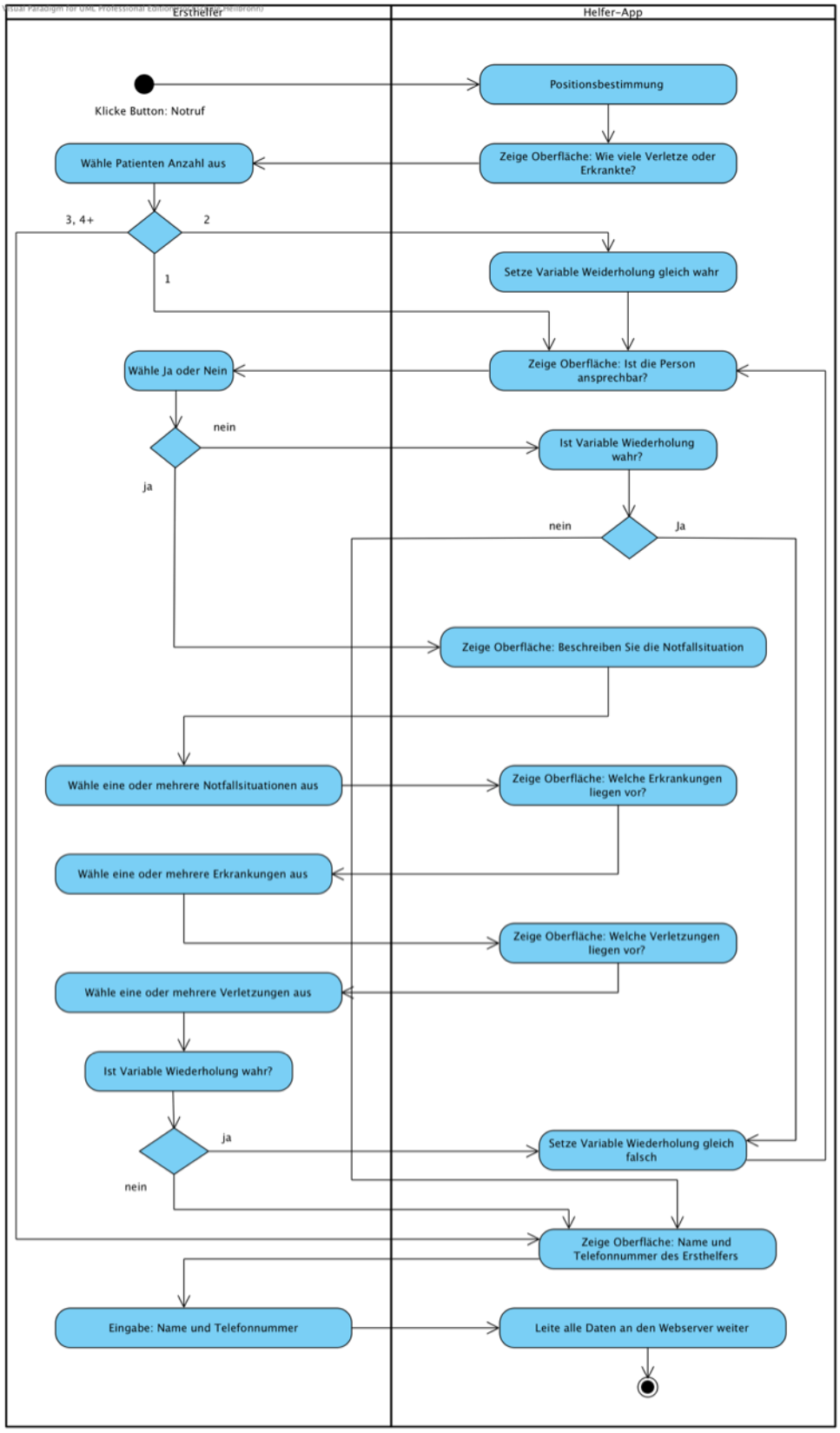


Abbildung 14: Aktivitätsdiagramm Alarmieren der Sanitäter

5.2.3 Use Case: Leitstelle anrufen

Sollte eine Voraussetzung für den Anwendungsfall *Alarmieren der Sanitäter* nicht gegeben sein oder hat der Ersthelfer keine Bestätigung erhalten, hat er mit der Funktion *Leitstelle anrufen* dennoch die Möglichkeit, medizinisches Fachpersonal anzufordern.

Vorbedingungen:

Für diesen Use Case benötigt der Ersthelfer ein Smartphone, welches in einer Funkzelle angemeldet sein muss. Mit der Funktion 112 startet der Anwender den Anwendungsfall.

Use Case:

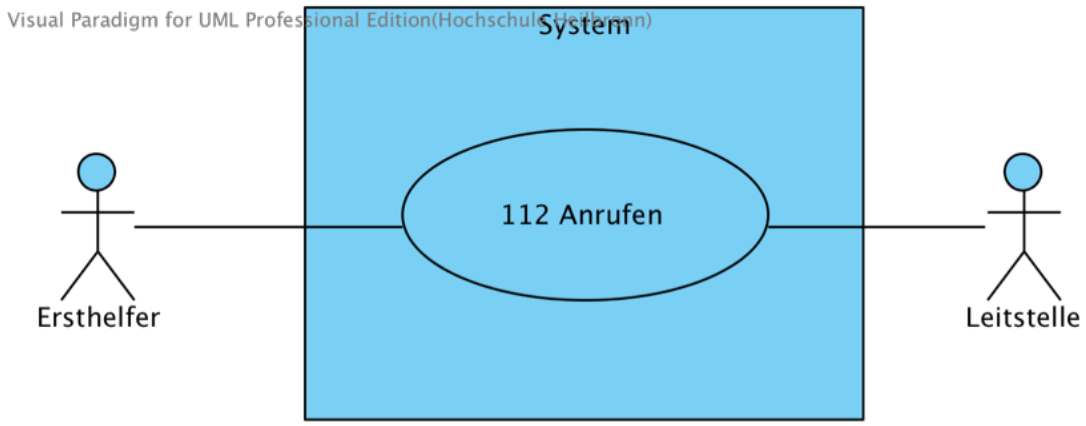


Abbildung 15: Use Case-Diagramm 112 anrufen

Beschreibung:

In diesem Use Case baut das Smartphone eine Verbindung mit der zuständigen Leitstelle auf. Über das Telefon kann der Ersthelfer die Informationen des Notrufes mündlich an den Korrespondenten der Leitstelle weiterleiten.

Nachbedingungen:

Der Korrespondent hat den Notruf entgegen genommen.

5.2.4 Use Case: Navigation zum Notausgang

Vorbedingungen:

Als Voraussetzung für diesen Anwendungsfall, muss der Ersthelfer ein iBeacon kompatibles Endgerät mit eingeschaltetem Bluetooth besitzen. Zudem muss er sich in der Reichweite eines iBeacons befinden. Der Ersthelfer startet den Anwendungsfall mit der Funktion Exit.

Use Case:

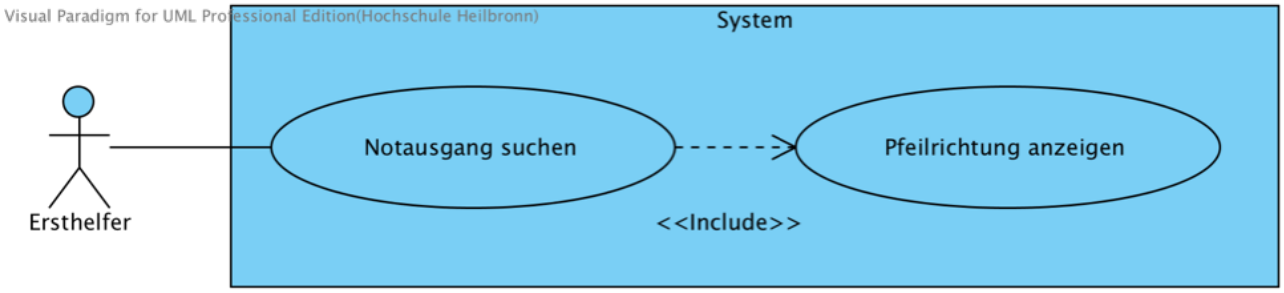


Abbildung 16: Use Case-Diagramm Navigation zum Notausgang

Beschreibung:

Bei diesem Anwendungsfall wird eine Navigation zum Notausgang durchgeführt. Dabei wird dem Ersthelfer ein Pfeil angezeigt, der in die Richtung des nächsten Notausganges weist. Dazu wird fortlaufend die Position des Ersthelfers über die in der Reichweite befindlichen Beacons ermittelt. Dies ist erforderlich, da sich der Ersthelfer Richtung Notausgang bewegt und dabei eventuell Umwege durch Hindernisse, wie Wände in Kauf nehmen muss. Durch die aktuelle Position des Ersthelfers und des Notausganges, wird der Pfeil ständig neu justiert.

Nachbedingungen:

Der Ersthelfer hat einen Notausgang erreicht.

5.3 Beschreibung der Sanitäter-App

Die Sanitäter-App hat die Anwendungsfälle Anzeigen und Aktualisieren der Patientenliste, Hinzufügen einer Notiz, Bestätigen eines Notrufs, sowie Navigation zum Patienten. Im Folgenden werden diese Anwendungsfälle näher beschrieben.

5.3.1 Use Case: Anzeigen der Notrufliste

Vorbedingungen:

Als Voraussetzung für diesen Anwendungsfall benötigen die Sanitäter ein Endgerät mit einer aktiven Internetverbindung. Der Use Case startet durch das Drücken des Notrufliste-Buttons durch den Sanitäter.

Use Case:

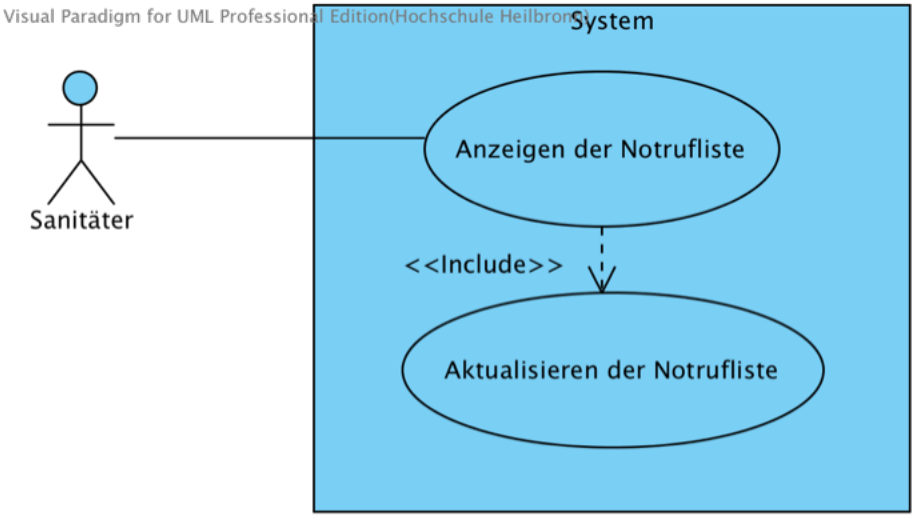


Abbildung 17: Use Case-Diagramm Anzeigen der Notrufliste

Beschreibung:

In diesem Use Case wird die Oberfläche für die Notrufliste geöffnet. Der Use Case beinhaltet den Use Case *Aktualisieren der Notrufliste*. Wird dieser nicht ausgeführt, bleibt die angezeigte Notrufliste leer.

Nachbedingungen:

Über das Auswählen eines Notrufes hat der Sanitäter die Möglichkeit, die Informationen der zum Notruf gehörigen Patienten einzusehen.

5.3.2 Use Case: Aktualisieren der Notrufliste

Vorbedingungen:

Als Voraussetzung für diesen Use Case muss die Notrufliste geöffnet sein. Des Weiteren ist eine aktive Internetverbindung erforderlich.

Use Case:

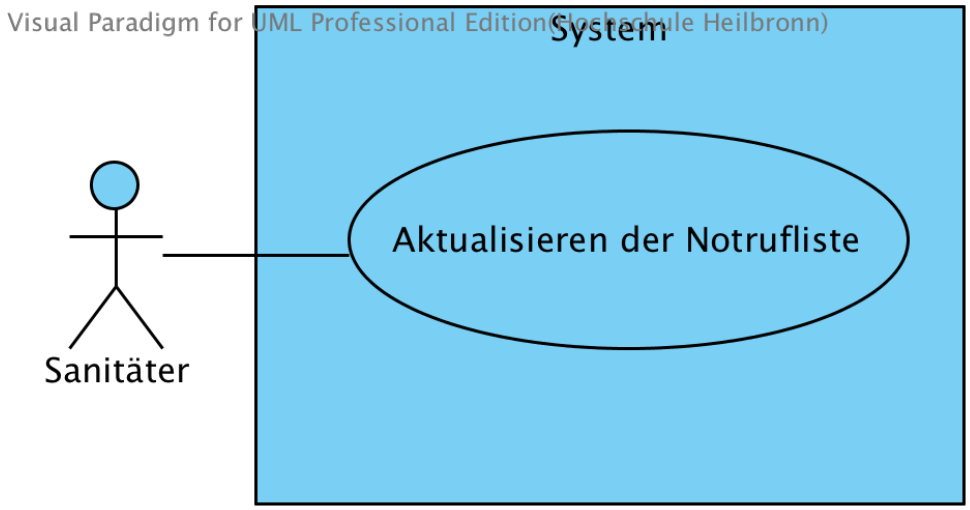


Abbildung 18: Use Case-Diagramm Aktualisieren der Notrufliste

Beschreibung:

Mit Hilfe dieses Anwendungsfalles wird die Liste mit den Notrufen aktualisiert. Dies geschieht zum einem automatisch nach dem Starten der App und ist zum anderen durch die Sanitäter, nach dem Eintreffen neuer Notrufe, auszuführen.

Nachbedingungen:

Die Notrufliste wurde aktualisiert.

5.3.3 Use Case: Bestätigen eines Notrufs

Vorbedingungen:

Als Vorbedingung für diesen Anwendungsfall muss eine aktive Internetverbindung vorhanden sein. Zudem muss ein unbestätigter Notruf in der Notrufliste vorhanden sein.

Use Case:

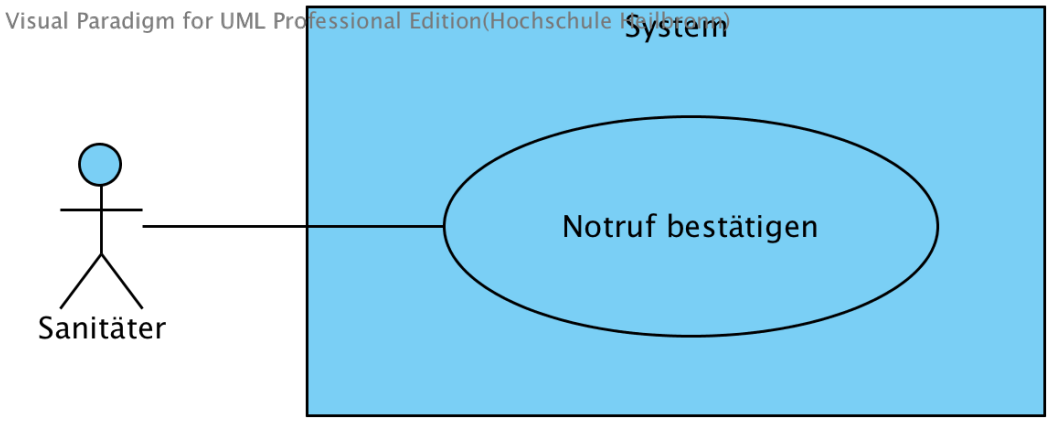


Abbildung 19: Use Case-Diagramm Notruf bestätigen

Beschreibung:

Durch das Drücken des Button mit dem Text *Notruf bestätigen*, können die Sanitäter einen Notruf bestätigen.
Das System dokumentiert das aktuelle Datum, sowie die aktuelle Zeit. Beides wird dem Notruf hinzugefügt.

Nachbedingungen:

Der Webserver hat die Information über die Bestätigung des Notrufs erhalten.

5.3.4 Use Case: Hinzufügen einer Notiz

Vorbedingungen:

Als Voraussetzung für diesen Use Case muss ein Notruf aus der Notrufliste ausgewählt werden.

Use Case:

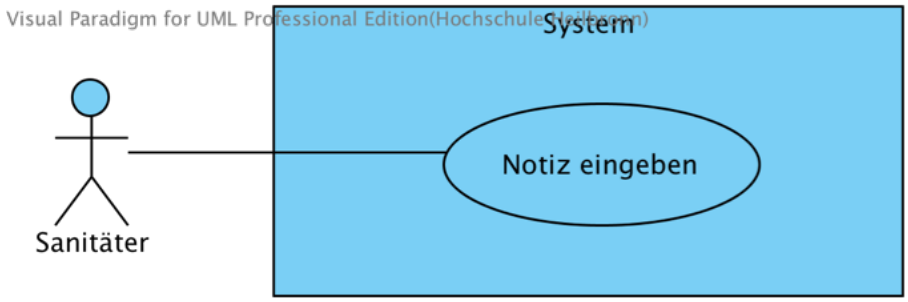


Abbildung 20: Use Case-Diagramm Notiz eingeben

Beschreibung:

Der Sanitäter kann zu jedem Patienten eines Notrufes private Notizen hinzufügen. Sollten vier oder mehr Patienten zum jeweiligen Notruf angehören, hat der Sanitäter die Möglichkeit, vier verschiedene Notizblöcke zu verwenden.
Zu der eingegebenen Notiz werden das aktuelle Datum sowie die aktuelle Uhrzeit dokumentiert.

Nachbedingungen:

Nach dem Eintragen der Notiz zeigt die App alle Notizen des ausgewählten Patienten bzw. des Notizblockes an.

5.3.5 Use Case: Navigation zum Patienten

Vorbedingungen:

Für diesen Use Case ist ein iBeacon kompatibles Endgerät erforderlich. Des Weiteren muss ein Notruf vorhanden sein, für den eine gültige Position durch die Helfer-App ermittelt wurde. Der Notruf muss zuvor über die Notrufliste aus dem Internet geladen und angezeigt werden.

Use Case:

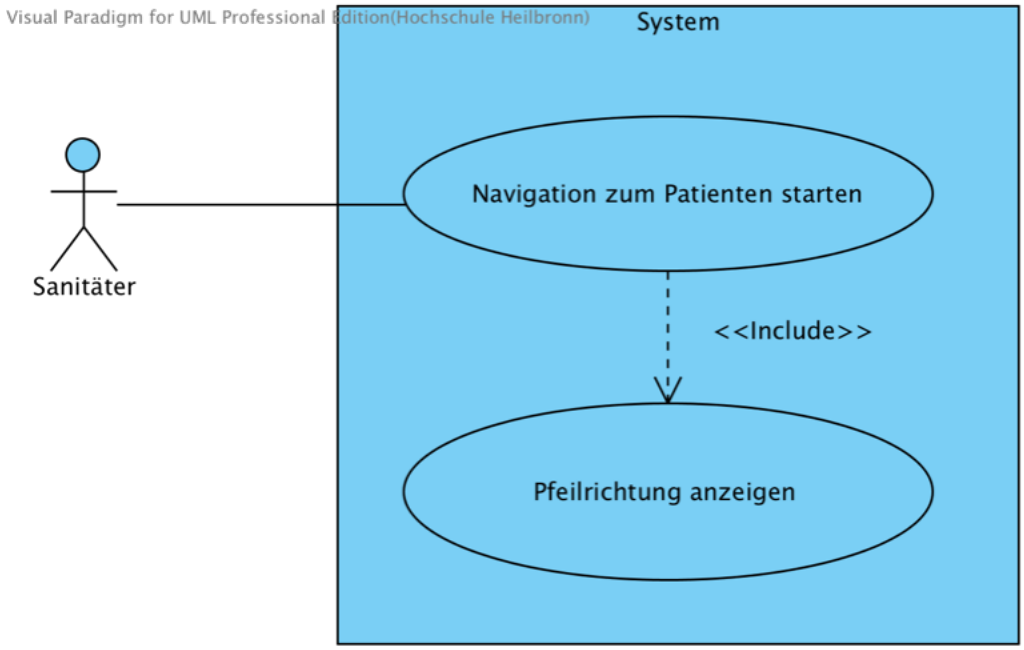


Abbildung 21: Use Case-Diagramm Navigation zum Patienten

Beschreibung:

Bei diesem Anwendungsfall wird eine Navigation zum gespeicherten Ort des Notrufs durchgeführt. Dabei wird den Sanitätern ein Pfeil angezeigt, der in die Richtung des Notruforts weißt. Dazu wird fortlaufend die Position der Sanitäter über die in der Reichweite befindlichen Beacons ermittelt. Dies ist erforderlich, da sich die Sanitäter Richtung Notrufort bewegen und dabei eventuell Umwege durch Hindernisse, wie Wände umgehen müssen. Durch die aktuelle Position der Sanitäter und die des Notruforts wird der Pfeil ständig neu justiert.

Nachbedingungen:

Die Sanitäter sind am Notfallort angekommen.

5.3.6 Use Case: Erstellen einer Radio Map

Vorbedingung:

Für diesen Use Case ist ein iBeacon kompatibles iPad mit aktivem Bluetooth erforderlich. Der Sanitäter muss sich mit dem iPad in dem Raum befinden, für den die Radio Map erstellt werden soll. Zudem muss die Größe des Raums angepasst werden.

Use Case:

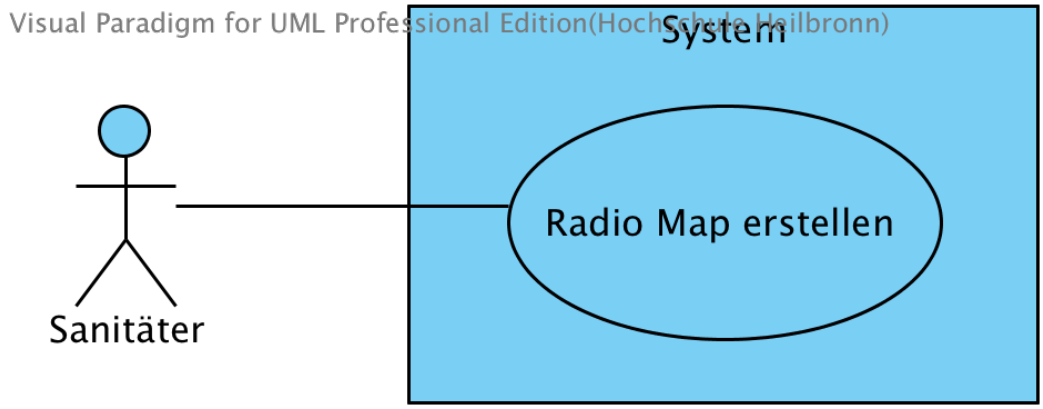


Abbildung 22: Use Case-Diagramm Radio Map erstellen

Beschreibung:

Die Sanitäter haben die Möglichkeit eine Radio Map zu erstellen, die für das Fingerprinting-Verfahren in der Helfer- und Sanitäter-App verwendet werden kann. Dazu stellt sich ein Sanitäter an die Position (0|0) des Koordinatensystems im Raum. Nachdem die Distanzen zu allen drei Beacons ermittelt wurden, geht der Sanitäter auf die nächste Position im Koordinatensystem z.B. (0|2) um dort ebenfalls die drei Distanzen zu den Beacons zu ermitteln. Dies wiederholt der Sanitäter für jede Position.

Nachbedingung:

Die Radio Map ist erstellt. Die Sanitäter müssen diese nun in die Helfer- und Sanitäter-App übertragen.

5.4 Architektur

Für den Betrieb des Prototyps sind zwei unabhängige Apps und ein Webserver, über den die beiden Apps kommunizieren können, erforderlich. Die Entwicklung der Apps erfolgt in der Programmiersprache Objective-C, da dies die primäre Programmiersprache für das iPhone bzw. das iPad ist. Die Apps werden für diese beiden Geräte optimiert, weil sie wie iBeacon von Apple entwickelt werden. Zudem eignet sich das iPad 4 aufgrund des großen Displays und der Kompatibilität mit dem Standard iBeacon für die Sanitäter-App. Das iPhone 5s, welches ebenfalls iBeacon kompatibel ist, eignet sich dagegen besonders durch das handliche Format für die Helfer-App.
Die Apps laufen zwar auf unterschiedlichen Endgeräten und werden von unterschiedlichen Benutzergruppen gebraucht, trotzdem sind Funktionen, wie die Navigation, sehr ähnlich. Als Architekturmuster empfiehlt sich deswegen das Model View Controller-Muster, dadurch können einzelne Komponenten für beide Apps verwendet werden.

5.4.1 Architektur der Helfer-App

Die Abbildung 23 zeigt das Klassendiagramm der Helfer-App. Es besteht aus vier Modulen. Die ersten Module *Model*, *View* und *Controller* entsprechen dem Model-View-Controller Prinzip. Das vierte Modul ist das *Framework* Modul.

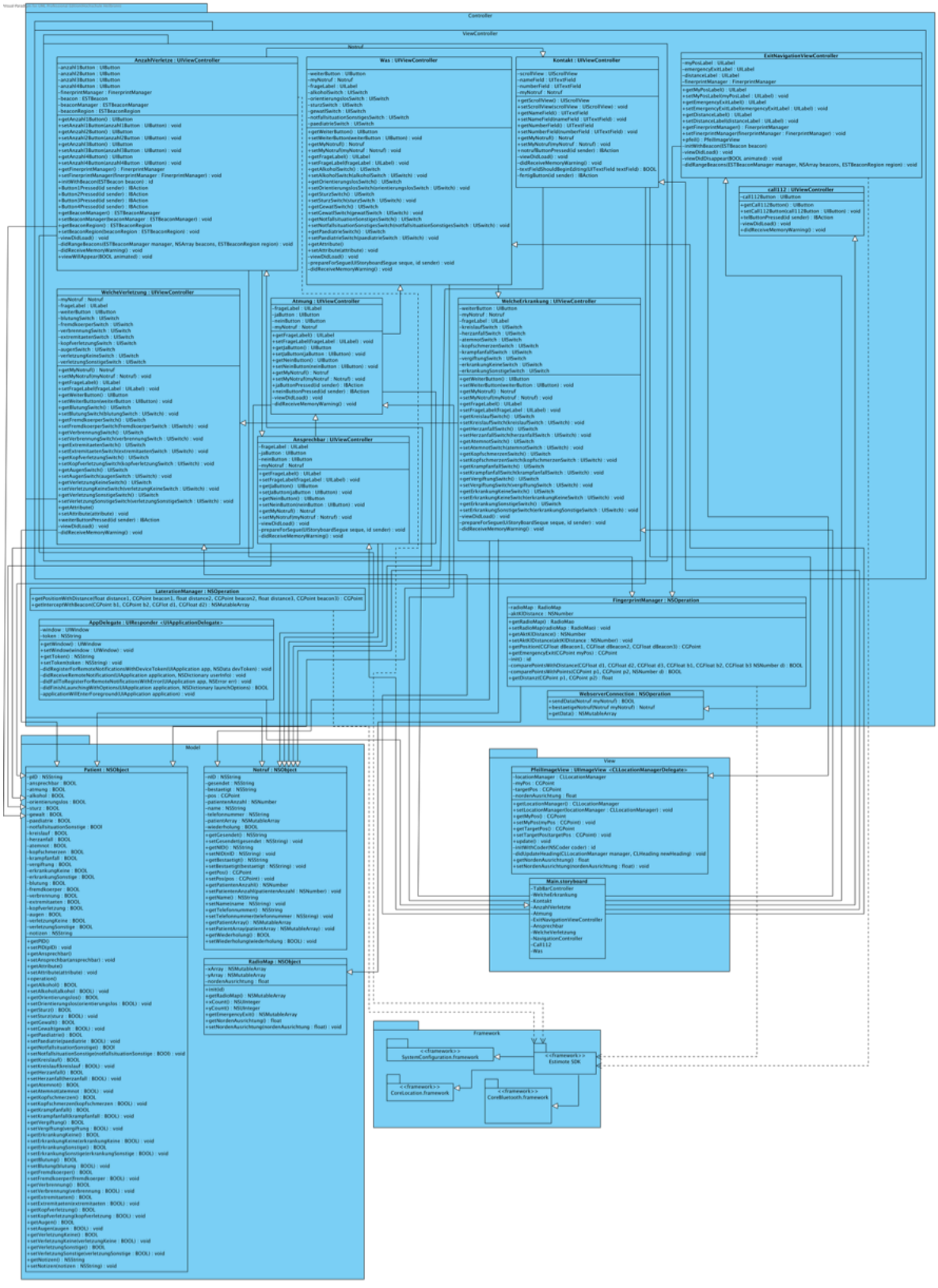


Abbildung 23: Architektur der Helfer-App

5.4.1.1 Modul Controller

Die Abbildung 24 zeigt das Modul *Controller*. Das Modul beinhaltet die Klassen *AppDelegate*, *LaterationManager*, *FingerprintManager* und *WebserverConnection*. Zusätzlich befinden sich im Modul *Controller* das Modul *ViewController* mit den Klassen *ExitNavigation* und *Call112*. Des Weiteren beinhaltet das Modul *ViewController* das Modul *Notruf* mit den Klassen *AnzahlVerletze*, *Ansprechbar*, *Atmung*, *Was*, *WelcheErkrankung*, *WelcheVerletzung* und *Kontakt*.

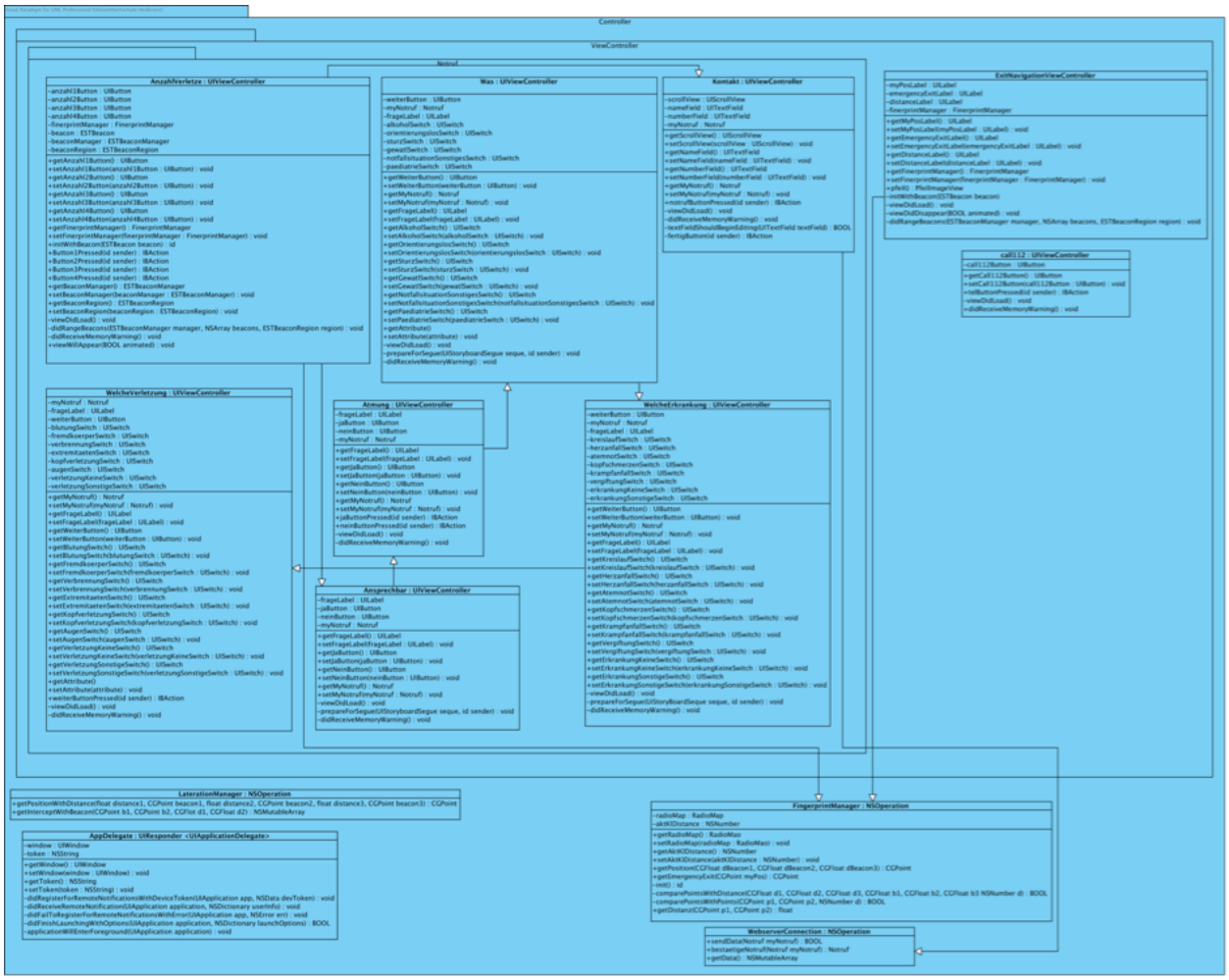


Abbildung 24: Modul Controller

Durch die Klasse *AppDelegate* werden die eintreffenden Push Notifications von der App entgegengenommen und verarbeitet. Zudem stellt sie der Klasse *WebserverConnection* das Device Token zur Verfügung.
Die Klasse *LaterationManager* misst die Abstände zu den Beacons und errechnet über das Verfahren der Lateration die Position des Endgerätes. Das Verfahren wird für die Tests in Kapitel 4 verwendet.

Alternativ zur Klasse *LaterationManager,* kann die Position ebenfalls über die Klasse *FingerprintManager* durch das Fingerprinting-Verfahren berechnet werden. Dieses Verfahren wird in der Helfer-App angewendet. Für das Fingerprinting-Verfahren erstellt die Klasse *FingerprintManager* ein Objekt der Klasse *RadioMap* mit der aktuellen Radio Map. Die Distanzen vom Endgerät zu den Beacons erhält die Klasse *FingerprintManager* von der Klasse *AnzahlVerletzte*, wenn die Sanitäter alarmiert werden sollen oder von der Klasse *ExitNavigation*, wenn der Ersthelfer die Navigation zum Notausgang starten möchte.
Die Klasse *WebserverConnection* ist für die Kommunikation zwischen Helfer-App und Webserver zuständig. Sie übermittelt den Notruf an den Webserver.
Die Klasse *ExitNavigation* erstellt ein Objekt der Klasse *FingerprintManager*, um die Position des Endgeräts zu ermitteln. Anschließend sucht die Klasse *FingerprintManager* nach dem nächsten Notausgang. Die Klasse *ExitNavigation* erstellt darauf ein Objekt der Klasse *PfeilImageView* und richtet den Navigationspfeil zum nächsten Notausgang aus.
Die Klasse *Call112* nimmt die Nutzerinteraktion, Drücken des Button *112 anrufen*, entgegen und gibt den Befehl, die Telefonnummer 112 anzurufen an das Smartphone weiter.
Die Klassen im Modul *Notruf* verwalten die Benutzereingaben zum Absetzen eines Notrufes. Jedes Objekt der Klassen besitzt dabei ein Objekt der Klasse *Notruf* und die passende Anzahl von Objekten der Klasse *Patient*.

5.4.1.2 Modul View

Das Modul *View* beinhaltet die Klasse *Main.Storyboard* in dem die Views zu den Klassen aus dem Modul ViewController erstellt werden. Zudem enthält das Modul die Klasse *PfeilImageView*. Diese richtet den Pfeil nach der von der Klasse *ExitNavigation* vorgegebenen Richtung aus.

Abbildung 25: Modul View

5.4.1.3 Modul Model

Das Modul *Model* beinhaltet drei Klassen: die Klassen *Notruf*, *Patient* und *RadioMap*. Die Klassen *Notruf* und *Patient* verwalten den Notruf mit der jeweiligen Anzahl von Patienten. Die Klasse *RadioMap* erstellt eine Radio Map und beinhaltet zudem eine Liste mit allen Notausgängen.

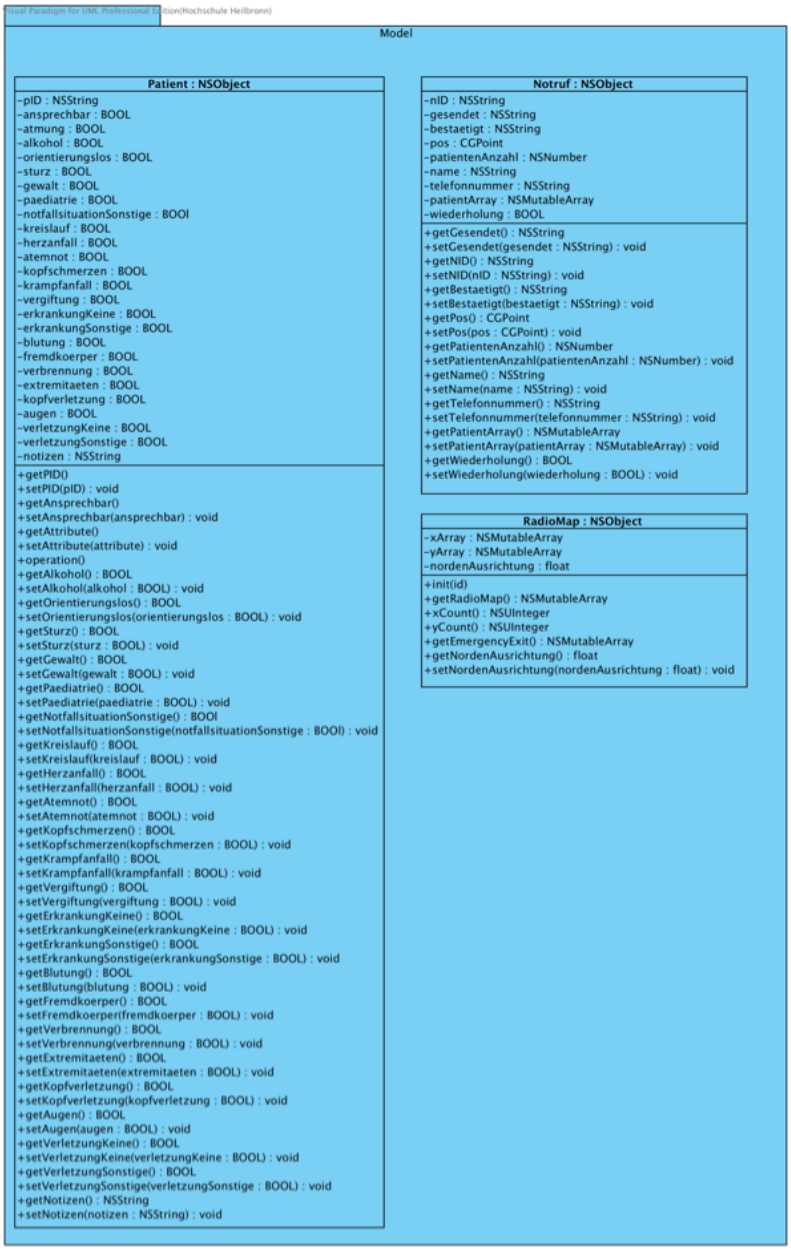


Abbildung 26: Modul Model

5.4.1.3 Modul Framework

Das Modul *Framework* beinhaltet das Framework *EstimoteSDK*, welches die Verbindung mit den Estimote Beacons ermöglicht. Das Framework *EstimoteSDK* benötigt die Frameworks *CoreLocation*, *SystemConfiguration* und *CoreBluetooth*, die sich ebenfalls in dem Modul *Framework* befinden.

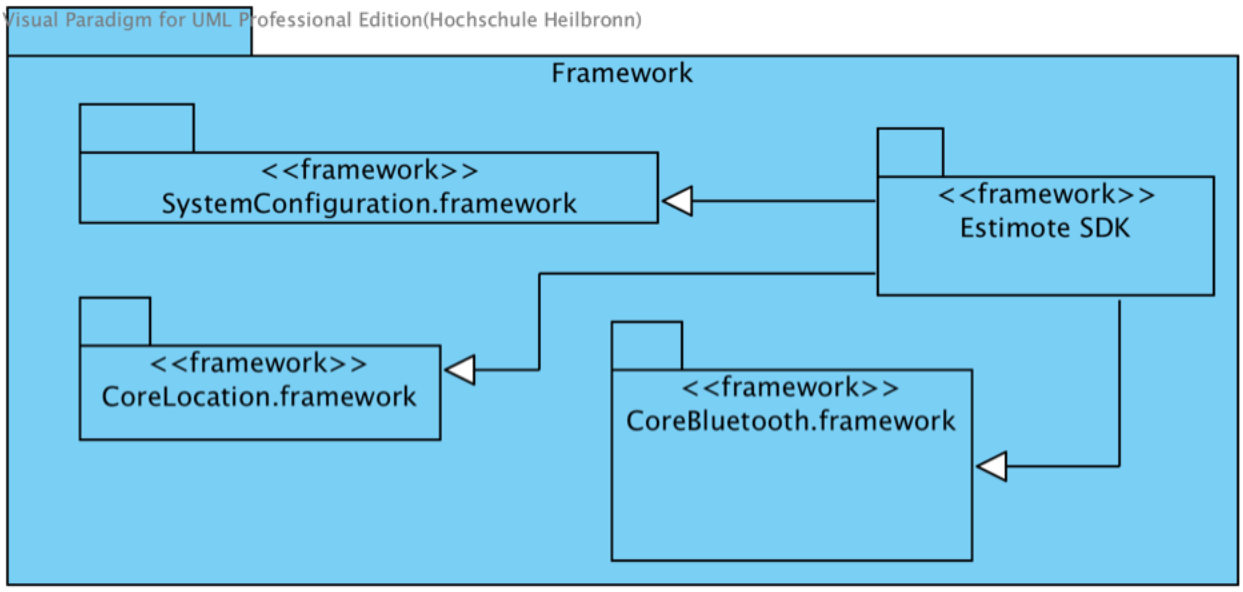


Abbildung 27: Modul Framework

5.4.2 Architektur der Sanitäter-App

Die Sanitäter-App besteht, wie die Helfer-App aus vier Modulen. Sie basieren ebenfalls auf dem Model-View-Controller Prinzip und das vierte Modul ist ebenfalls das *Framework* Modul.

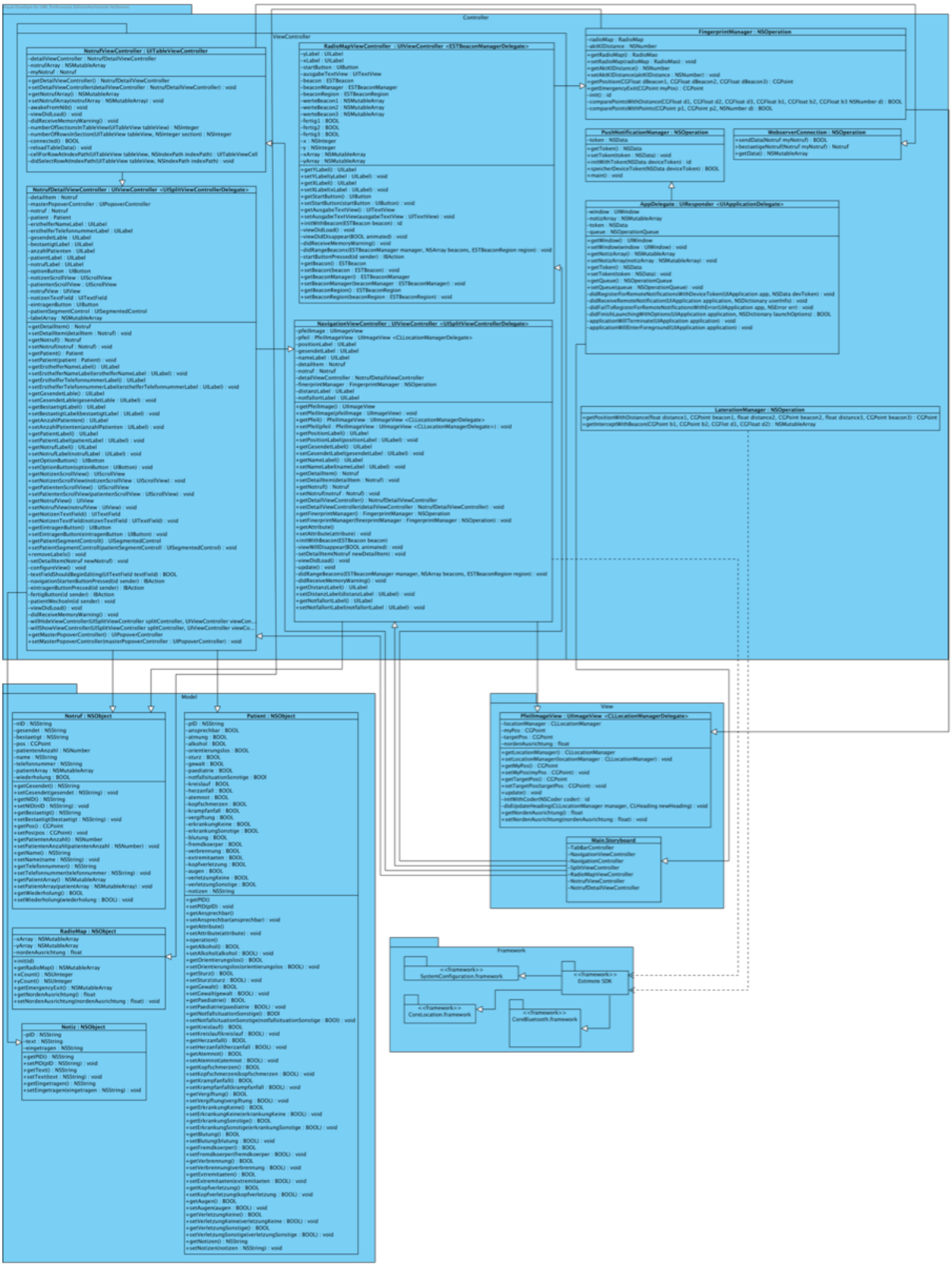


Abbildung 28: Architektur der Sanitäter-App

5.4.2.1 Modul Controller

Das Modul Controller besteht aus den Klassen *AppDelegate*, *PushNotificationManager*, *WebserverConnection*, *FingerprintManager*, *LaterationManager* und dem Untermodul *ViewController*. Dies beinhaltet die Klassen *NotrufViewController*, *NotrufDetialViewController*, *NavigationViewController* und *RadioMapViewController*.

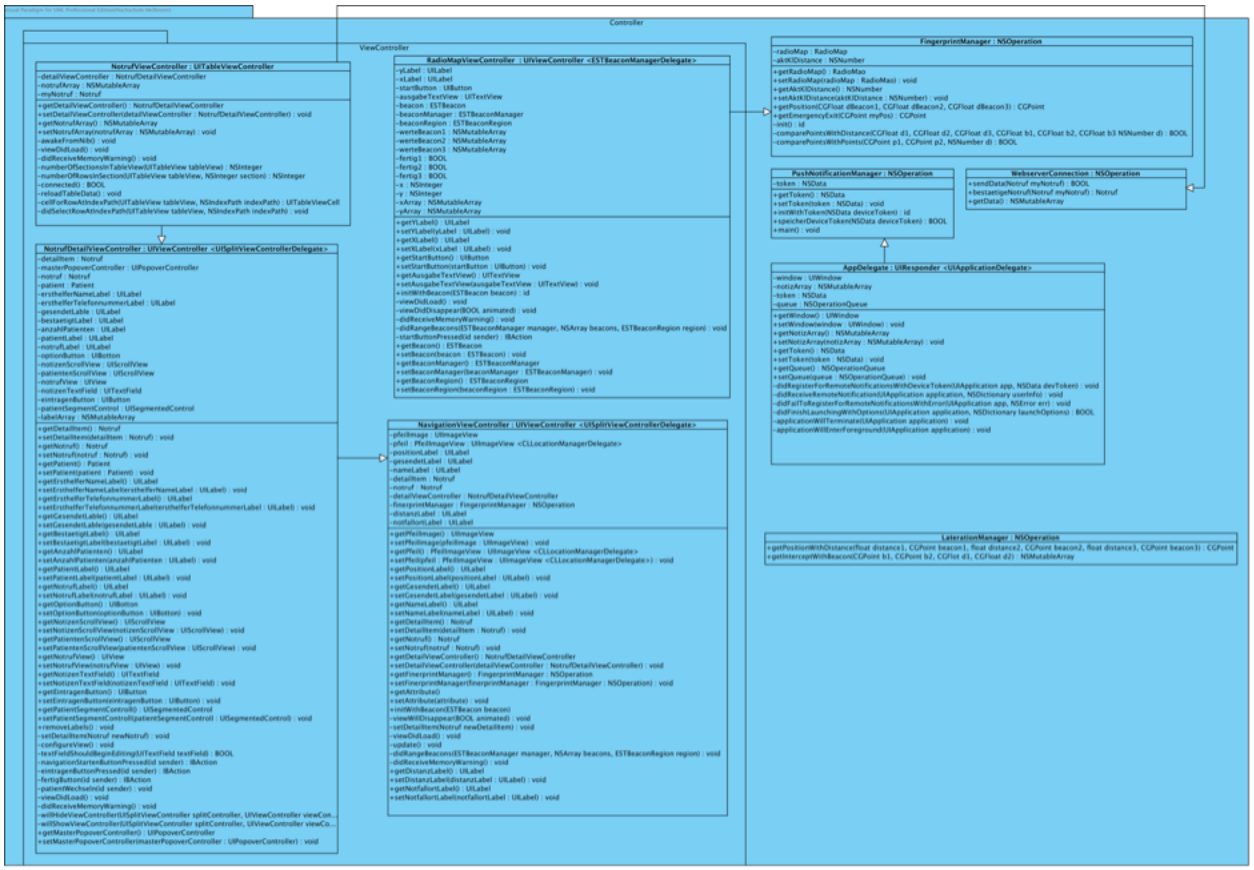


Abbildung 29: Modul Controller

Die Klasse *PushNotificationManager* übermittelt den Device Token des Endgerätes an den Webserver. Durch die Klasse *AppDelegate* werden die eintreffenden Push Notifications von der App entgegengenommen und verarbeitet. Zudem stellt die Klasse *AppDelegate* dem *PushNotificationManager* das Device Token zur Verfügung. Des Weiteren stellt die Klasse *AppDelegate sicher*, dass die Notizen der Sanitäter zu einem Notruf gespeichert werden, wenn die App geschlossen wird.

Die Klasse *WebserverConnection* ist für die Synchronisation der Notrufinformationen zwischen Webserver und Sanitäter-App zuständig. Sie lädt ein JSON-Objekt mit den Notrufinformationen herunter und speichert diese Information in Objekte der Klassen *Notruf* und *Patient*. Zudem teilt die Klasse *WebserverConnection* dem Webserver mit, wenn ein Notruf durch die Sanitäter bestätigt wird.
Die Klasse *LaterationManager* wird für die Tests in Kapitel 4 verwendet. Sie ermittelt mit Hilfe der Estimote SDK die Distanzen zwischen dem Endgerät und den Beacons. Durch die gemessenen Distanzen bestimmt die Klasse *LaterationManager* nach dem Verfahren der Lateration die Position des Endgerätes.

Die Klasse *FingerprintManager* wird von der Sanitäter-App zur Positionsbestimmung verwendet. Die Klasse *FingerprintManager* erhält von der Klasse *NavigationViewController* die Distanzen des Endgerätes zu den Beacons. Mit Hilfe dieser Distanzen und der Klasse *RadioMap*, errechnet die Klasse *FingerprintManager* die aktuelle Position des Endgerätes. Die Klasse *NavigationViewController* gibt diesen Standort und den Notrufort an die Klasse *PfeilImageView* weiter, die den Pfeil zur Navigation ausrichten.
Die Klasse *NotrufViewController* ist ein Objekt der Klasse *UITableViewController* und zeigt eine Liste mit den Notrufen an. Durch Auswählen eines Notrufs wird ein Objekt der Klasse *NotrufDetailViewController* gebildet, welches die Klasse *UISplitViewControllerDelegate* implementiert und die Informationen zum ausgewählten Notruf anzeigt. Das Objekt der Klasse *NotrufDetailViewController* erstellt wiederum ein Objekt der Klasse *NavigationViewController*.
Die Klasse *RadioMapViewController* erstellt eine Radio Map und gibt die Distanzen zu den Beacons für jede gemessene Position aus.

5.4.2.2 Modul View

Das Modul *View* der Sanitäter-App besteht aus der Klasse *PfeilImageView* und dem *Main.storyboard*.

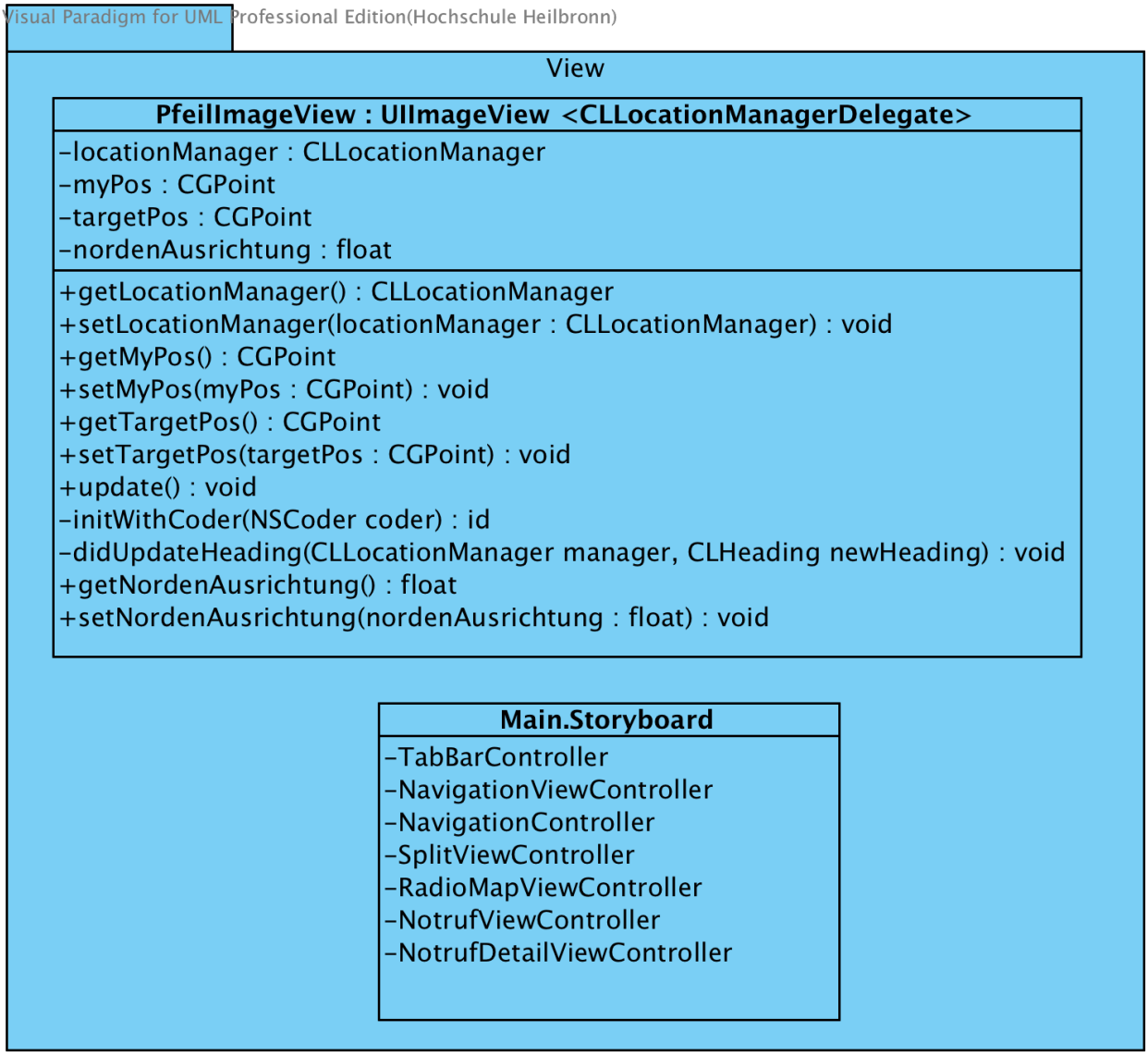


Abbildung 30: Modul View

Die Klasse *PfeilImageView* wird durch die Klasse *NavigationViewController* erstellt. Die Klasse *PfeilImageView* richtet den Pfeil mit den Positionen des Endgerätes und des Notfallorts aus.
In der Klasse *Main.Storyboard* werden die Views zu den Klassen aus dem Modul ViewController erstellt.

5.4.2.3 Modul Model

Das Modul *Model* beinhaltet die Klassen *Notruf*, *Patient*, *RadioMap* und *Notiz*

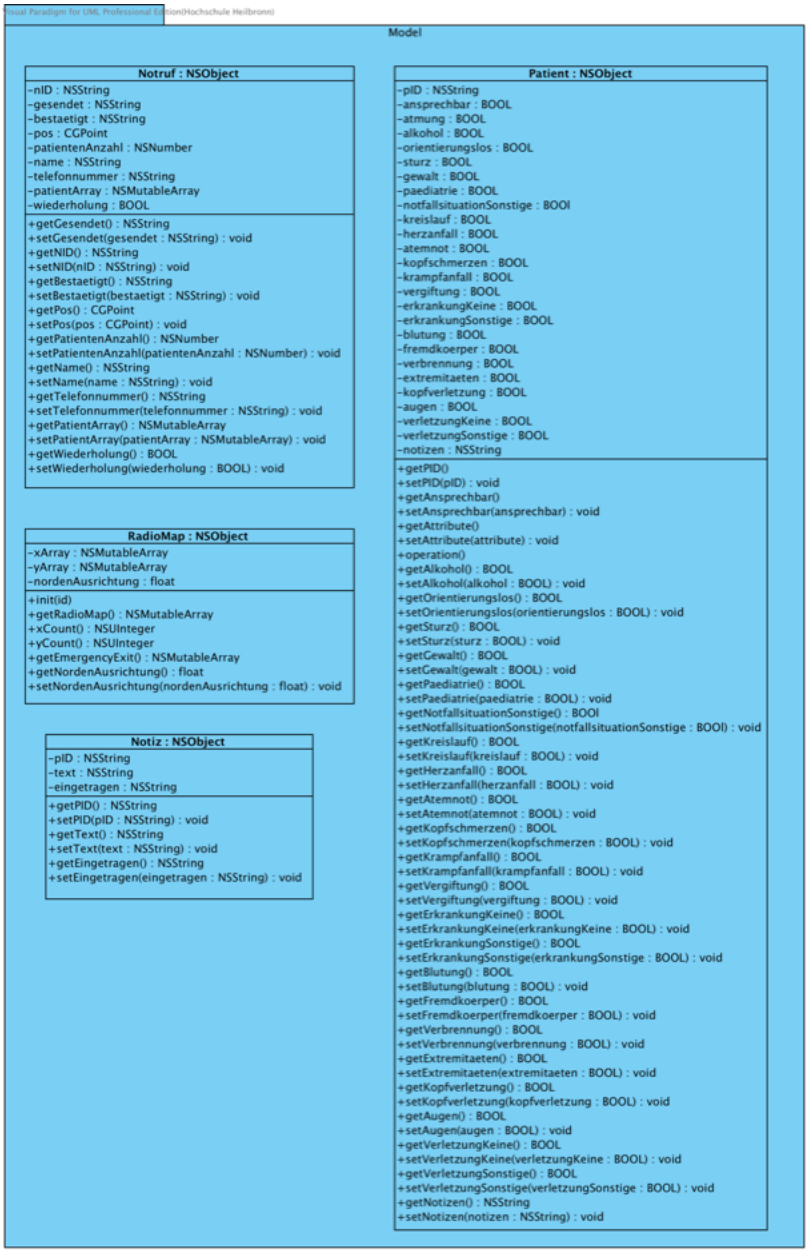


Abbildung 31: Modul Model

Wie auch in der Helfer-App beinhaltet die Klasse *RadioMap* die von der Klasse *FingerprintManager* verwendeten Informationen zu der Radio Map. Die Klassen *Notruf* und *Patient* verwalten die in der Klasse *WebserverConnection* erstellten Notrufe mit den dazu gehörigen Patienten. Die Klasse *Notiz* speichert die von dem Sanitäter zu den Patienten hinzugefügten Notizen. Dokumentiert werden die Notiz und die Zeit, sowie das Datum des Eintrages.

5.4.3 Kommunikation zwischen Helfer- & Sanitäter-App

Die Kommunikation zwischen den beiden Apps regelt ein Webserver. Dieser ist zum einem für das Versenden der Benachrichtigung (Push Notification) über neue Notfälle und deren Bestätigung durch die Sanitäter notwendig. Zum anderen speichert der Webserver die Informationen des Notrufs in einer Datenbank. So sind die Daten persistent gespeichert und lesbar durch verschiedene Endgräte.

5.4.3.1 Versenden der Push Notification

Die Abbildung 32 beschreibt den Ablauf für das Versenden einer Push Notification. Die Schritte 1-3 werden dabei nur einmal ausgeführt, wobei hingegen die Schritte 4-5 bei jedem Versenden einer neuen Push Notification ausgeführt werden.

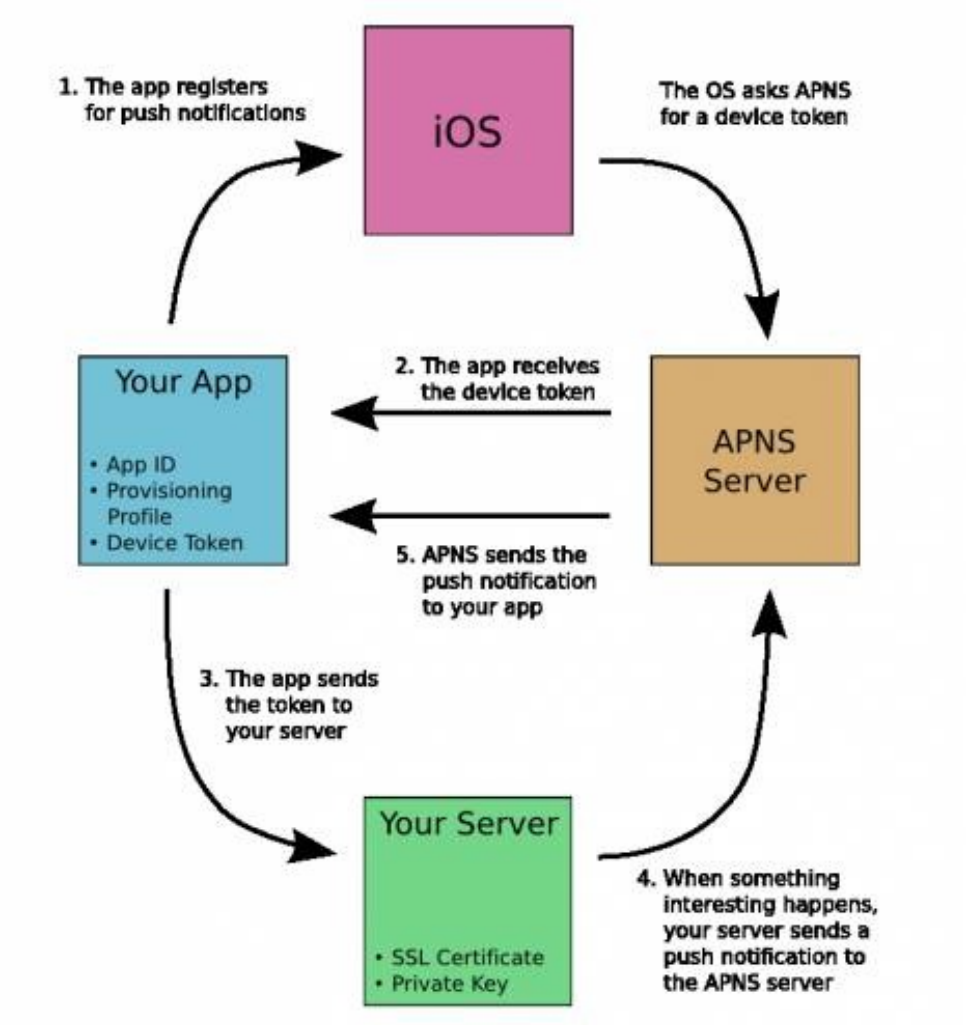


Abbildung 32: Ablauf zum Versenden einer Push Notification [31]

Im ersten Schritt fragt die App den Benutzer, ob dieser eine Push Notification von der App empfangen möchte. Stimmt der Benutzer zu, wird der Device Token abgefragt. Jedes Endgerät hat zwei verschiedene Device Token. Einen für die Entwicklung (sandbox) und einen für den Betrieb (production). Da es sich bei den Apps um einen Prototypen handelt, wird der sandbox-Device Token verwendet. Die App fragt den Apple Push Notification service Server (APNs) nach dem Device Token. Dieser leitet den Token an die App weiter, welche ihn wiederum an den Webserver weitergibt (Schritt 3). Anschließend wird der token in einer MySQL Datenbank gespeichert. Die Device Token der
Ersthelfer werden dabei dem jeweiligen Notruf angehängt, um eine spätere Zuordnung zu ermöglichen. Die Device Token der Sanitäter werden in einer eigenen Datenbank gespeichert.
Setzt der Benutzer von der Helfer-App einen Notruf ab, leitet die App den Befehl, eine Push Notification zu versenden über den Webserver an den Apple Push Notification service Server weiter (Schritt 4). Der APNs Server versendet schließlich im fünften Schritt eine Push Notification an alle registrierten Endgeräte der Sanitäter-App.
Bestätigt ein Sanitäter einen Notruf, leitet die Sanitäter-App den Befehl, eine Push Notification zu versenden, von der Sanitäter-App über den Webserver an den APNs weiter (Schritt 4). Der APNs Server versendet eine Push Notification an das Gerät, von dem der Notruf abgesetzt wurde.

5.4.3.2 Speichern der Notrufinformationen

Zur Speicherung der Notrufinformationen übermittelt die App diese mit Hilfe des Hypertext Transfer Protocol (HTTP) über die Post-Methode an den Webserver. Dieser baut eine Verbindung zur MySQL Datenbank auf und speichert die Informationen des Notrufs persistent ab.
Die Abbildung 33 zeigt den Aufbau dieser Datenbank. Dabei ist jeder Patient genau einem Notruf zugeordnet. Dadurch kann ein Notruf mehrere Patienten besitzen.

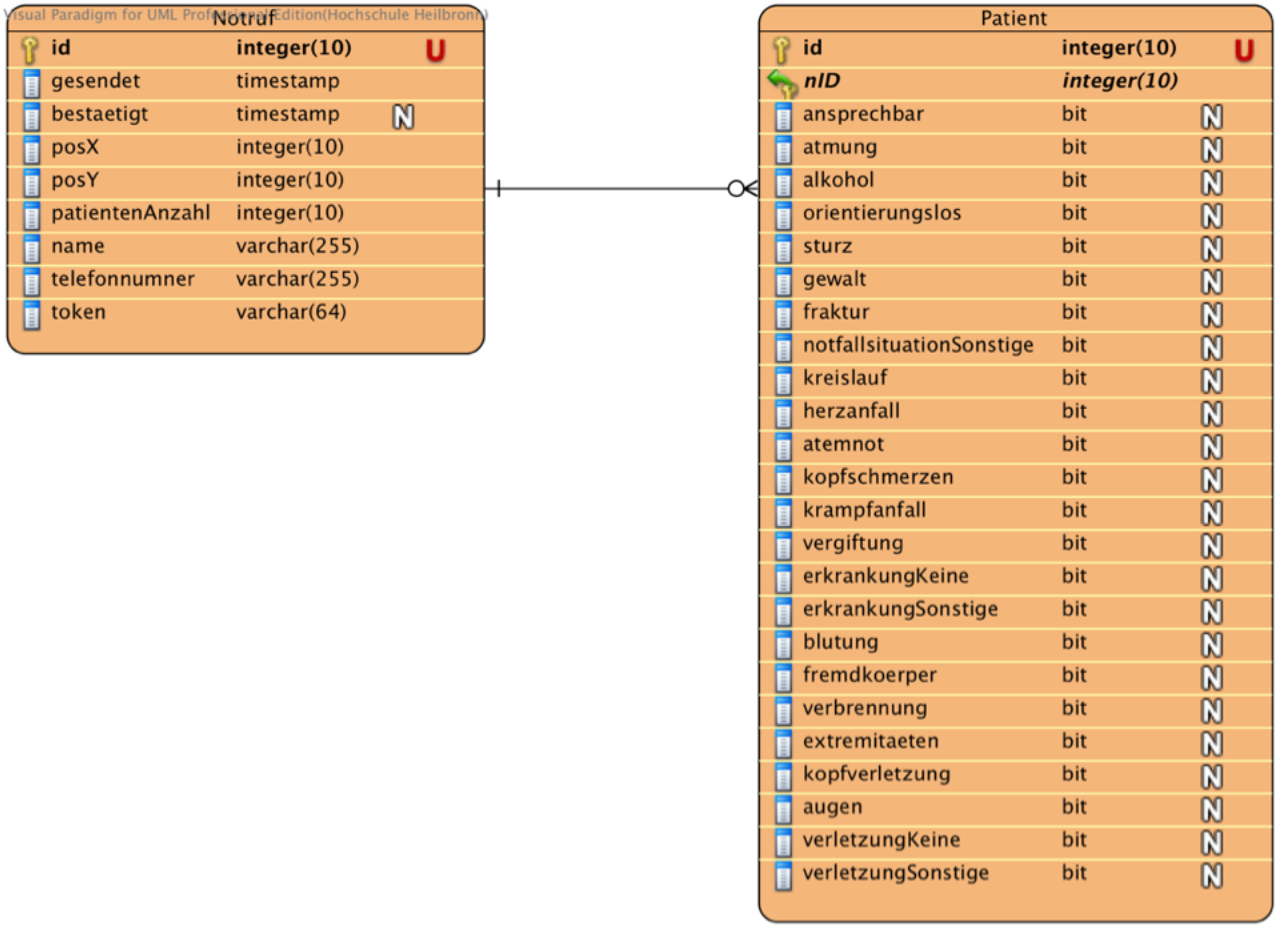


Abbildung 33: Aufbau der Datenbank

5.4.3.3 Abrufen der Notrufinformationen

Der Webserver stellt die Notrufinformationen, welcher in einer MySQL Datenbank gespeichert sind, in Form eines JavaScript Object Notation (JSON) - Objekts online zur Verfügung.
Jedes Gerät, auf dem die Sanitäter-App installiert ist, hat die Möglichkeit, dieses JSON-Objekt aufzurufen und die Informationen darzustellen.

5.4.3.4 Alarmieren der Sanitäter

Im Folgenden wird die Alarmierung der Sanitäter durch den Ersthelfer beschrieben. Als Beispiel wird das Szenario mit einem ansprechbaren Patienten gewählt.
Für die Alarmierung der Sanitäter wird in der Klasse *AnzahlVerletzte* ein Objekt der Klasse *Notruf* erstellt. Wählt der Ersthelfer die Anzahl der Patienten aus, werden passend zu der ausgewählten Anzahl Objekte der Klasse *Patient* erstellt. In diesem Beispiel wird ein Objekt *Patient* erstellt. Das Objekt wird dem zuvor erstellten Objekt der Klasse *Notruf* angehängt, sodass im weiterem Verlauf nur das Objekt der Klasse *Notruf* weitergereicht werden muss. Die ausgewählte Patientenanzahl wird ebenfalls im Objekt *Notruf* gespeichert. Anschließend wird ein Objekt der Klasse *Ansprechbar* generiert. Diesem Objekt wird der *Notruf* mit dem *Patienten* angehängt. Es öffnet sich die Oberfläche *Ansprechbar*. Der Ersthelfer kann Ja oder Nein auswählen. In diesem Beispiel wählt er Ja aus, sodass ein Objekt der Klasse *Was* erstellt wird. Diesem Objekt wird das Objekt *Notruf* angehängt. Der Ersthelfer wählt nun aus, was passiert ist. Drückt der Ersthelfer auf den weiter-Button, werden die Informationen gespeichert und es wird ein Objekt der Klasse *WelcheErkrankungen* erstellt. Diesem Objekt wird ebenfalls das Objekt *Notruf* angehängt. Mit der neu erscheinenden Oberfläche wählt der Ersthelfer aus, welche Erkrankungen der Patient hat. Durch das Drücken des weiter-Buttons werden die Informationen im Objekt *Patient,* welches dem Objekt *Notruf* angehängt ist*,* gespeichert und es wird ein Objekt der Klasse *WelcheVerletzungen* erstellt. Diesem wird das Objekt *Notruf* angehängt. Der Ersthelfer kann mit Hilfe der neuen Oberfläche auswählen, welche Verletzungen der Patient hat. Diese Informationen werden beim Drücken des weiter-Buttons im Objekt *Patient* gespeichert. Es wird nun ein Objekt der Klasse *Kontakt* erstellt. Das Objekt *Notruf* wird dem neu erstellten Objekt angehängt. Hat der Ersthelfer bereits in der Vergangenheit einen Notruf gesendet, werden sein Name und seine Telefonnummer automatisch in die dafür vorgesehenen *UITextField* geladen. Andernfalls muss der Ersthelfer sie eintragen. Die Daten werden in diesem Falle für weitere Notrufe gespeichert. Drückt der Ersthelfer auf den Button mit dem Text *Notruf an die Sanitäter übermitteln*, wird ein Objekt der Klasse *WebserverConnection* erstellt. Anschließend wird die Methode *-(BOOL)sendData(Notruf *):myNotruf* aufgerufen. Diese sendet die Daten an den Webserver, welcher sie in einer Datenbank speichert. War die Speicherung erfolgreich, sendet der Webserver die Antwort *gesendet*. In diesem Fall liefert die Methode *-(BOOL)sendData(Notruf *):myNotruf* ein

true zurück. Dem Ersthelfer wird bestätigt, dass der Notruf übermittelt wurde.

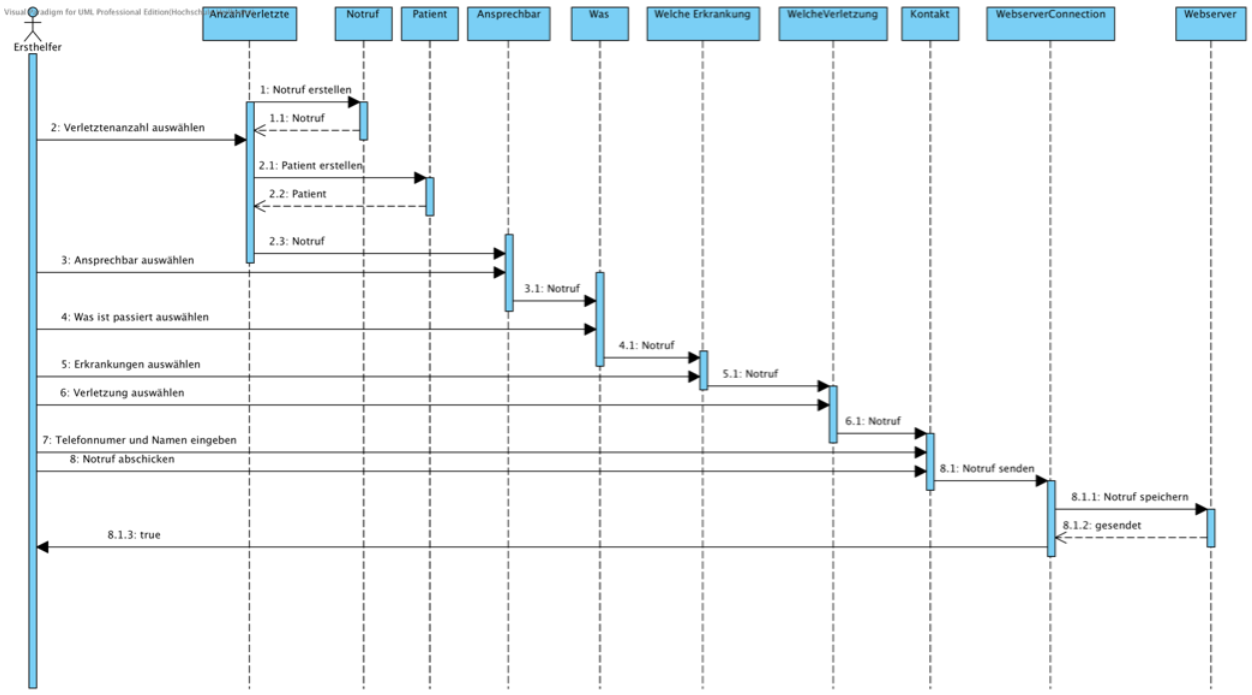


Abbildung 34: Sequenzdiagramm Alarmieren der Sanitäter

5.4.3.5 Bestätigung des Notrufs

Über die Klasse *NotrufDetailViewController* haben die Sanitäter die Möglichkeit, einen Notruf zu bestätigen. Bei einer Bestätigung wird die Klasse *WebserverConnection* aufgerufen, die Methode *bestaetigeNotruf(Notruf: notruf)* generiert darauf das aktuelle Datum und ruft anschließend die Datei *bestaetigeNotruf.php* auf dem Webserver auf. Diese aktualisiert das Bestätigungsdatum in der Datenbank. Zudem sendet *bestaetigeNotruf.php* eine Anfrage an den Apple Push Notification service (APNs) Server, welche eine Push Notification an den Ersthelfer sendet. Die Push Notification beinhaltet die Nachricht: „Ihr Notruf wurde bestätigt!“. Sind die Aktualisierung der Datenbank und das Versenden der Push Notification erfolgreich, liefert der Webserver eine *1* zurück. Die Klasse *WebserverConnection* gibt darauf den Notruf an die Klasse *NotrufDetailViewController* zurück. Die Klasse öffnet ein Alert mit dem Hinweis, dass die Bestätigung erfolgreich war. Zudem wird das Bestätigungsdatum angezeigt. Der Notruf färbt sich in der Notrufliste von blau zu schwarz. War die Bestätigung nicht erfolgreich, wird den Sanitätern ein Alert angezeigt, indem auf die erfolglose Bestätigung hingewiesen wird.

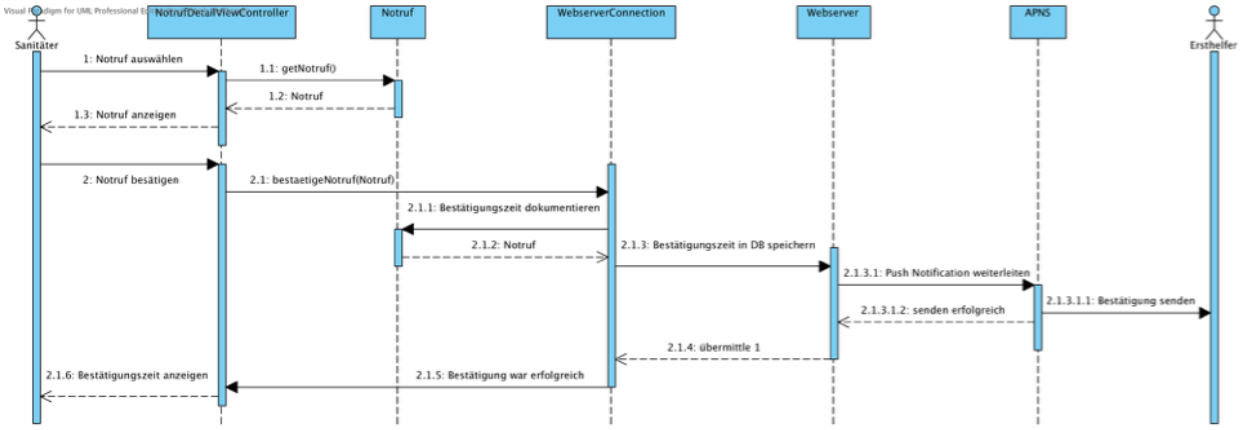


Abbildung 35: Sequenzdiagramm Bestätigung des Notrufs

5.4.3.6 Navigation zum Patienten

Über die Klasse *NotrufDetailViewController* können die Sanitäter die Navigation zum Patienten starten. Die Klasse *NotrufDetailViewController* übermittelt dazu den Notfallort an ein Objekt der Klasse *NavigationViewController*. Dieses berechnet wiederum die Distanzen vom Endgerät zu den einzelnen Beacons. Sobald die Distanzen ermittelt sind, werden sie an ein Objekt der Klasse *FingerprintManager* übergeben. Das Objekt errechnet mit Hilfe der Klasse *RadioMap* die Position des Endgeräts.
Das Objekt der Klasse *NavigationViewController* fragt nun bei der Klasse *FingerprintManager* nach der Ausrichtung der Radio Map nach Norden an. Die Klasse *FingerprintManager* erhält diese Information von der Klasse *RadioMap*. Das Objekt der Klasse NavigationViewController übermittelt nun die Position des Endgeräts, die Position des Ersthelfers (Notfallort) und die Ausrichtung der Radio Map nach Norden an ein Objekt der Klasse *PfeilImageView*. Dieses berechnet mit den drei Informationen und der Ausrichtung des Endgeräts nach Norden den Winkel, um den der Pfeil gedreht wird. Das Objekt der Klasse *PfeilImageView* zeigt dem Ersthelfer den Navigationspfeil an.

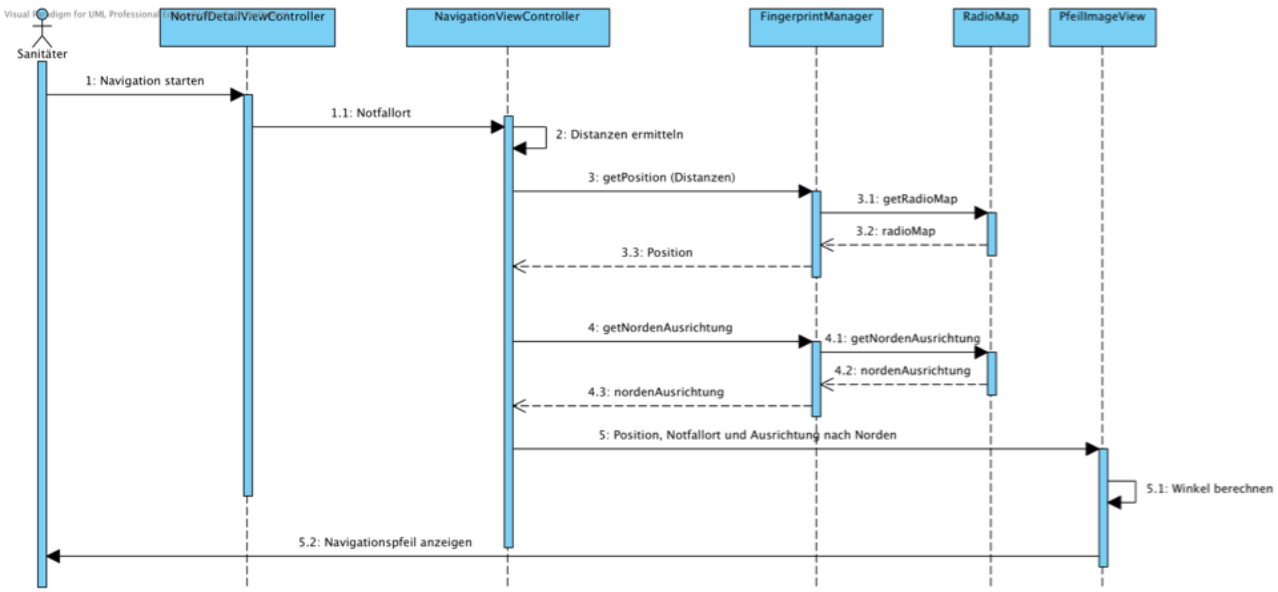


Abbildung 36: Sequenzdiagramm Navigation zum Patienten

5.4.3.7 Navigation zum nächsten Notausgang

Über ein Objekt der Klasse *ExitNavigationController* kann der Ersthelfer die Navigation zum nächsten Notausgang starten. Dazu ermittelt das Objekt die Distanzen zwischen dem Endgerät und den Beacons. Sobald alle Distanzen ermittelt sind, werden die Distanzen an ein Objekt der Klasse *FingerprintManager* übergeben. Dieses Objekt ermittelt mit Hilfe eines Objekts der Klasse *RadioMap*, welches die Radio Map liefert, die Position des Endgeräts. Anschließend übergibt das Objekt der Klasse *ExitNavigationController* die Position an die Klasse *FingerprintManager*, damit diese den nächsten Notausgang berechnen kann. Dazu ist eine Liste mit allen Notausgängen erforderlich. Diese erhält die Klasse *FingerprintManager* von der Klasse *RadioMap*. Sobald die Klasse *FingerprintManager* den nächsten Notausgang ermittelt hat, übergibt sie den Notausgang an das Objekt der Klasse *ExitNavigationController*. Das Objekt holt sich nun über die Klasse *FingerprintManager* bei der Klasse *RadioMap* die Ausrichtung der Radio Map nach Norden. Das Objekt der Klasse ExitNavigationController übergibt darauf an ein Objekt der Klasse *PfeilImageView* die Position des Endgeräts, die Position des nächsten Notausgangs und die Ausrichtung der Radio Map nach Norden. Das Objekt der Klasse *PfeilImageView* ermittelt mit diesen drei Werten und der Ausrichtung des Endgeräts nach Norden den Winkel, um den der Navigationspfeil gedreht wird. Das Objekt der Klasse *PfeilImageView* zeigt über den Navigationspfeil dem Ersthelfer den nächsten Notausgang an.

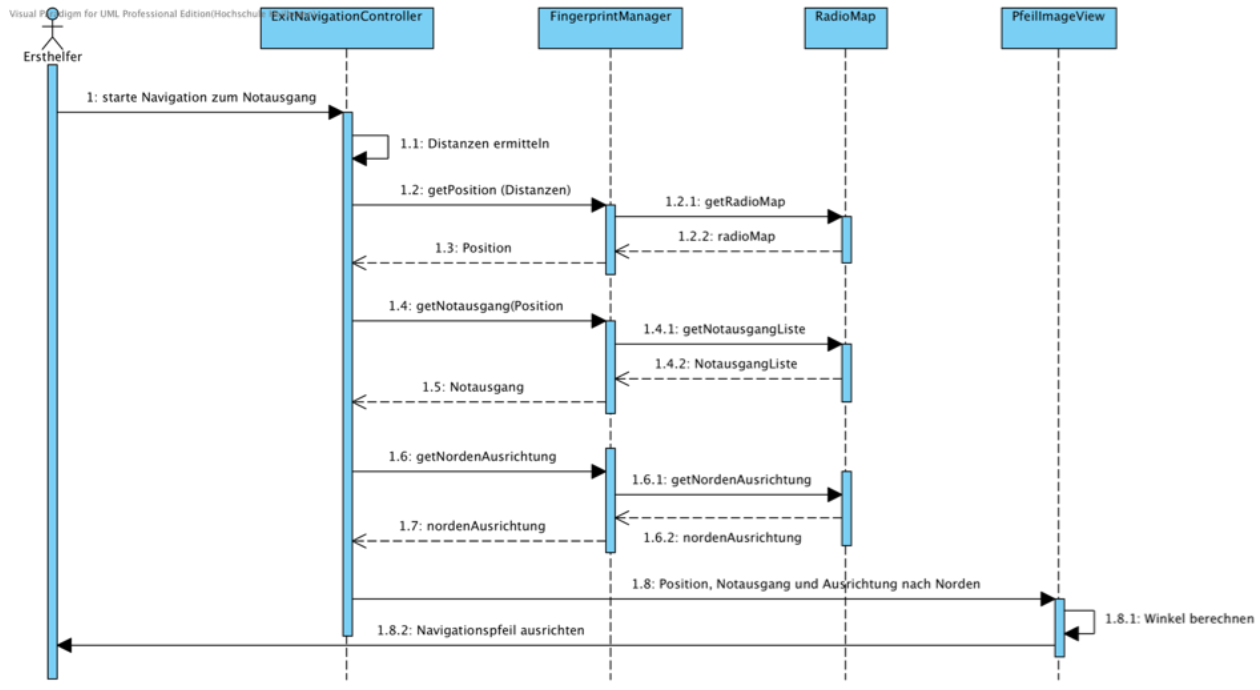


Abbildung 37: Sequenzdiagramm Navigation zum Notausgang

5.5 Entwurf der Benutzeroberfläche

In diesem Kapitel werden die Benutzeroberflächen der Helfer- und der Sanitäter-App beschrieben. Die Helfer-App wird dabei auf einem iPhone 5s und die Sanitäter-App auf einem iPad 4 ausgeführt.

5.5.1 Oberfläche der Helfer-App

Die Abbildung 38 zeigt die installierte Helfer-App auf einem iPhone.

Abbildung 38: Installierte Helfer-App

5.5.1.1 Alarmieren der Sanitäter

Über den Button *Notruf* hat der Ersthelfer die Möglichkeit, die Alarmierung der Sanitäter zu starten. Die zuerst angezeigte Benutzeroberfläche ermöglicht es dem Ersthelfer, durch das Drücken des jeweiligen Buttons, eine Patientenanzahl auszuwählen.

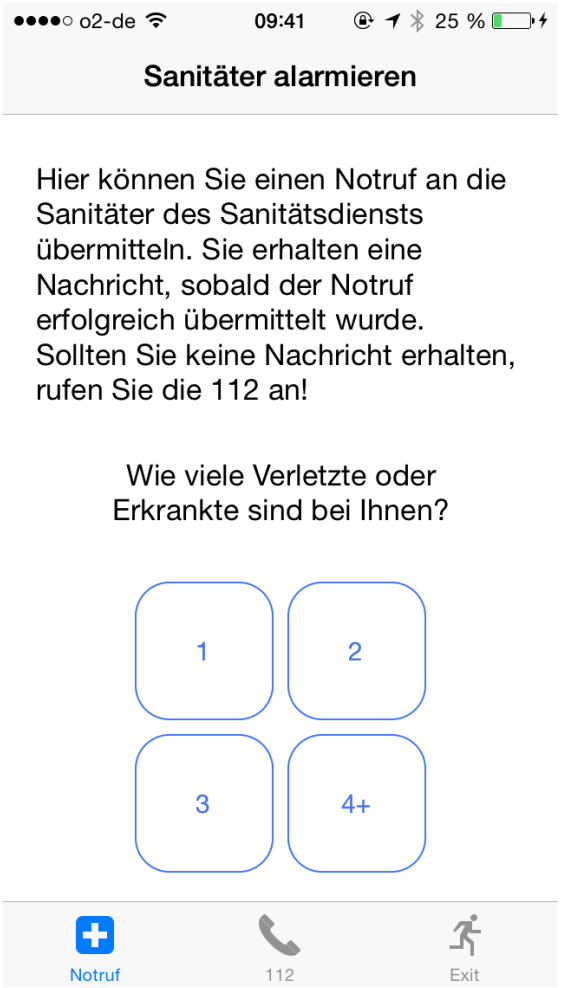


Abbildung 39: Auswahl der Patientenanzahl

Der Standort des Ersthelfers wird zeitgleich ermittelt. Ist die Ermittlung noch nicht abgeschlossen, kann keine Patientenanzahl ausgewählt werden. Sollte der Ersthelfer dennoch versuchen eine Patientenanzahl auszuwählen, erhält er eine Nachricht mit dem Hinweis zu warten. Wurde der Standort erfolgreich ermittelt, erhält der Ersthelfer ebenfalls eine Nachricht.

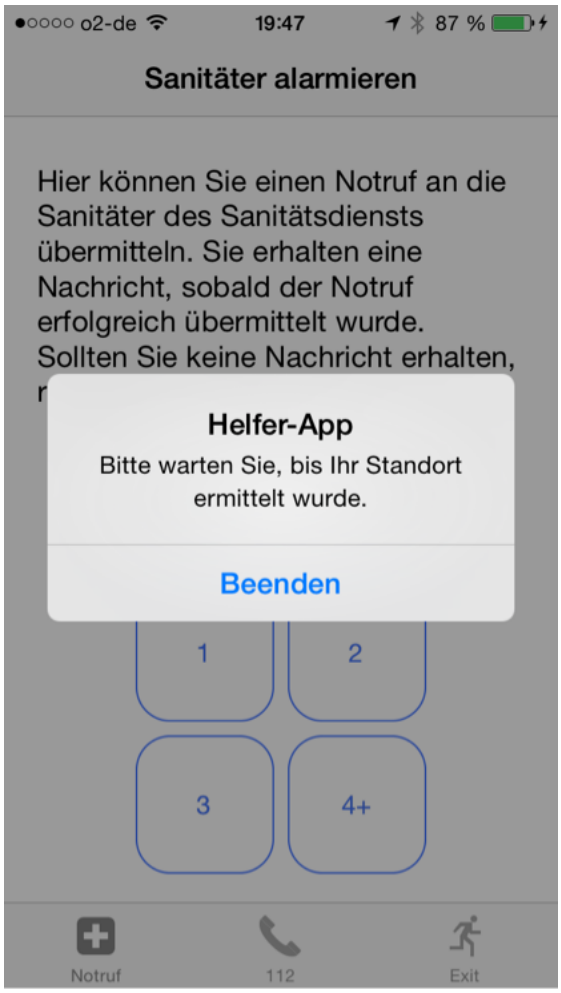
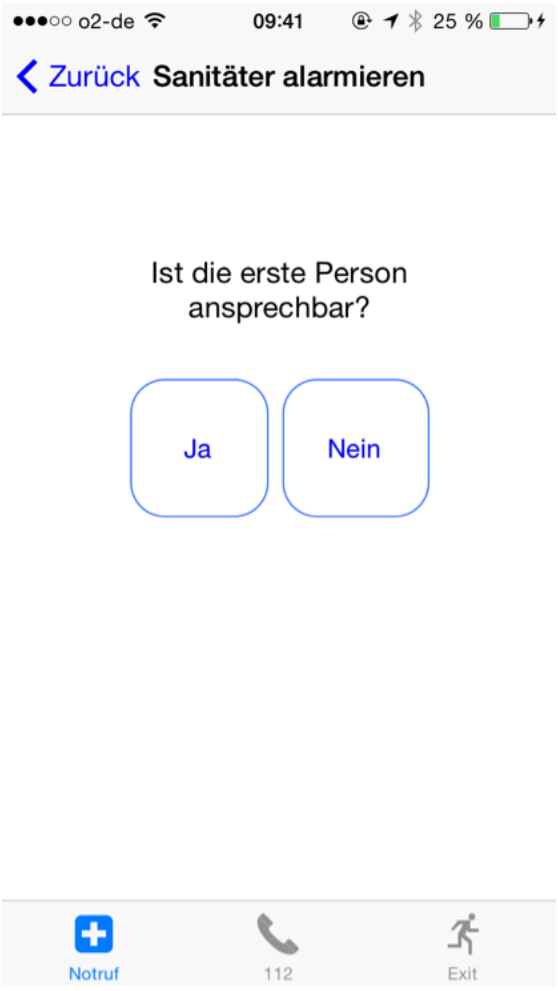
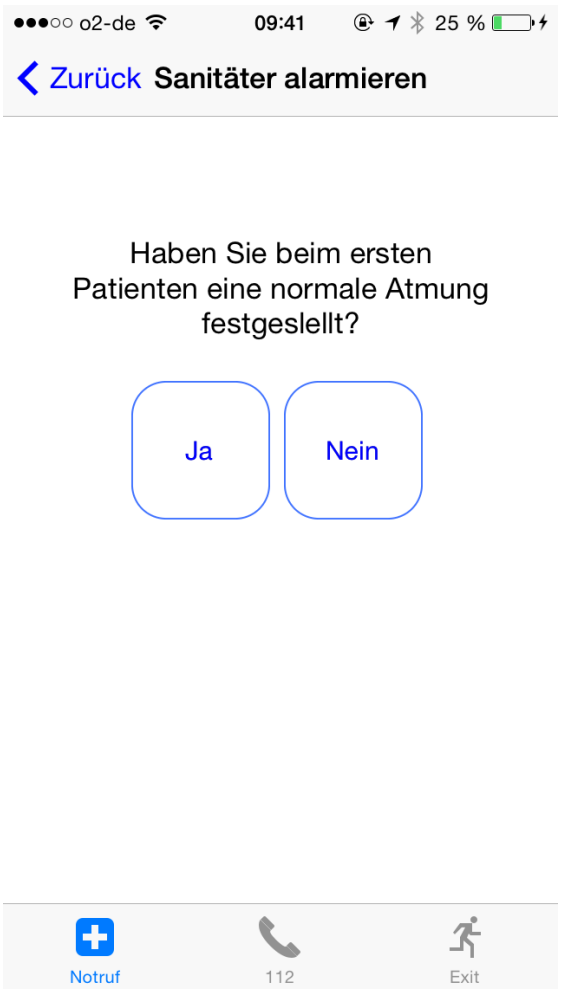


Abbildung 40: Standort wird ermittelt

Abbildung 41: Standort wurde ermittelt

Wie in Kapitel 5.5.2 beschrieben hat die Auswahl der Patientenanzahl Auswirkungen auf die nächste Oberfläche. Das nächste Beispiel zeigt die Oberflächen für zwei bewusstlose Patienten. Nach diesen Oberflächen wird die Oberfläche Kontakt angezeigt.





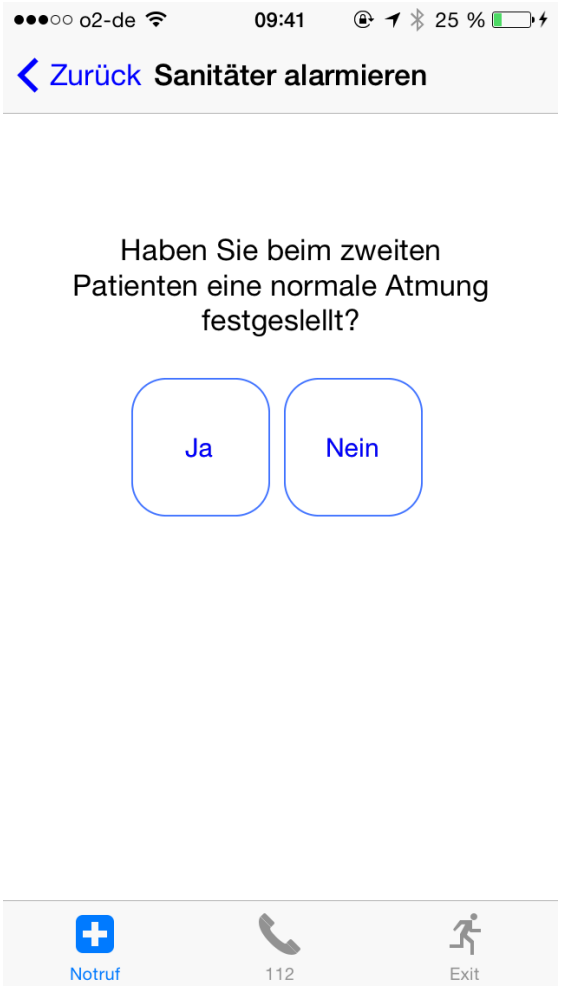


Abbildung 42-45: Auswahl Ansprechbar und normale Atmung

Beträgt die Patientenanzahl eins oder zwei und ist einer der Patienten ansprechbar, erfolgt eine detaillierte Abfrage zu der Notfallsituation, die Erkrankungen und Verletzungen des Patienten. Folgende Screenshots zeigen die Benutzeroberfläche zur Abfrage der Informationen für den ersten Patienten:

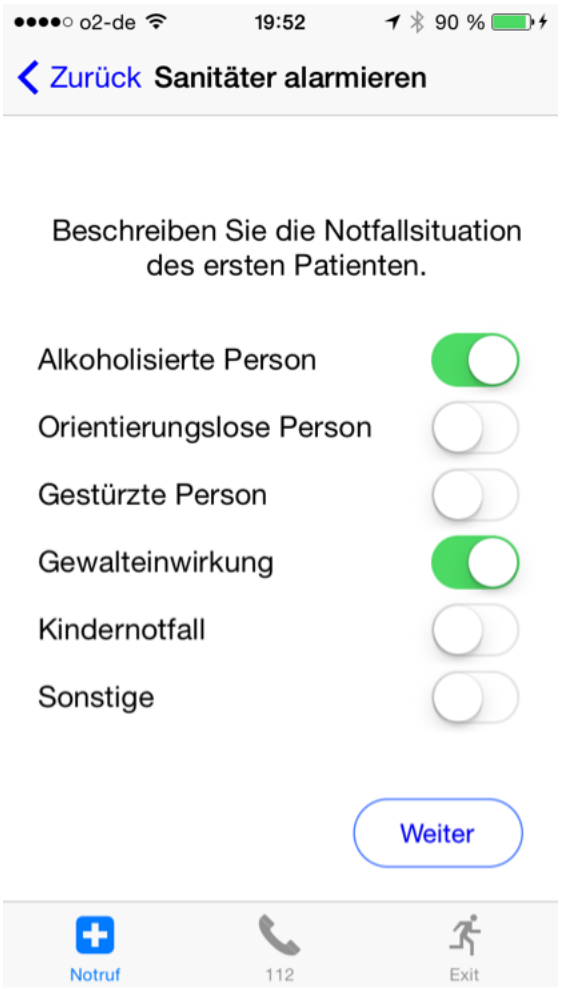


Abbildung 46: Auswahl der Notfallsituation

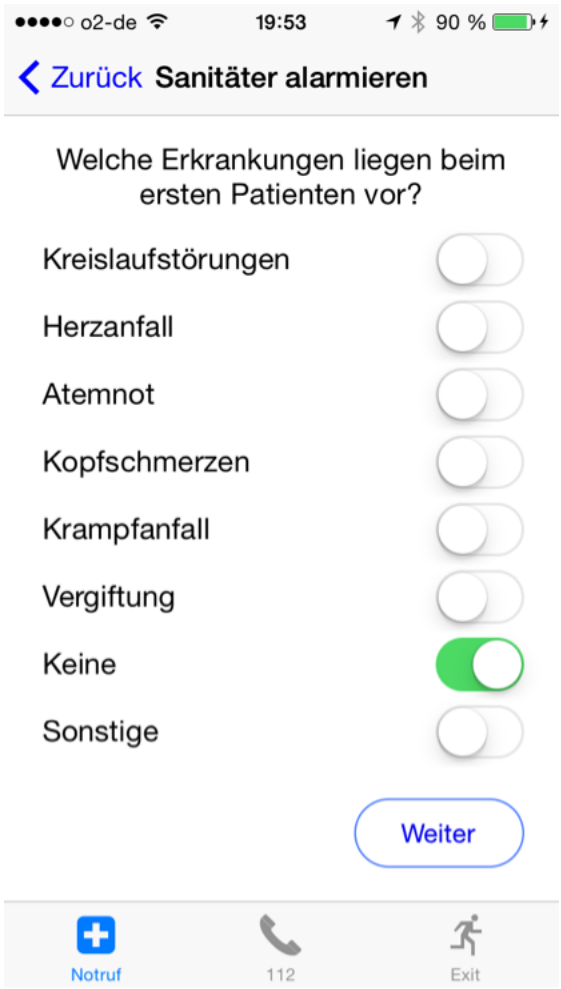


Abbildung 47: Auswahl der Erkrankungen

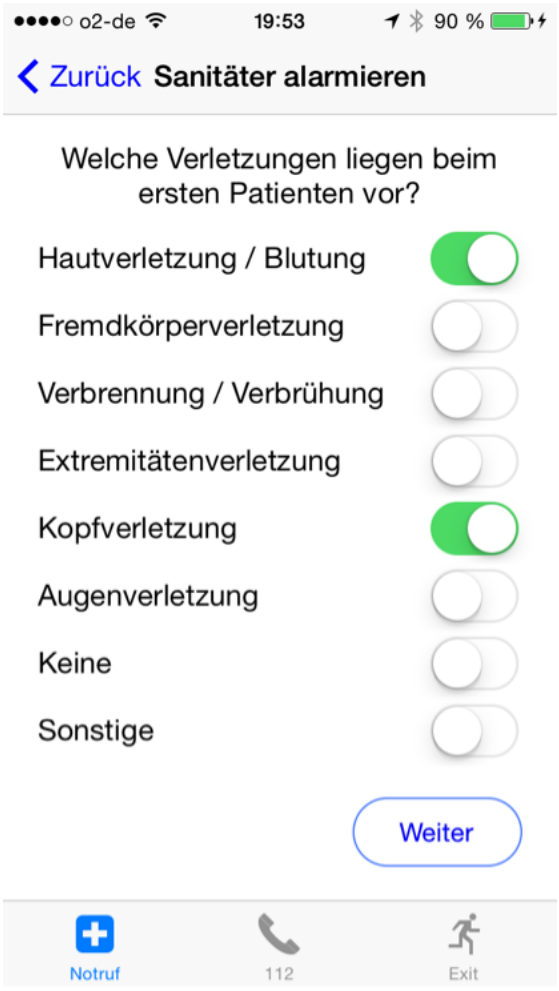


Abbildung 48: Auswahl der Verletzungen

Wurden alle Informationen zu den Patienten angegeben, wird der Ersthelfer aufgefordert seine Daten für Rückfragen anzugeben. Das Eingabefeld für den Namen des Ersthelfers, darf dabei nicht leer sein. Zudem muss die eingegebene Telefonnummer aus mehr als zwei Ziffern bestehen. Über folgende Oberflächen können die Daten eingegeben und der Notruf an die Sanitäter übermittelt werden:

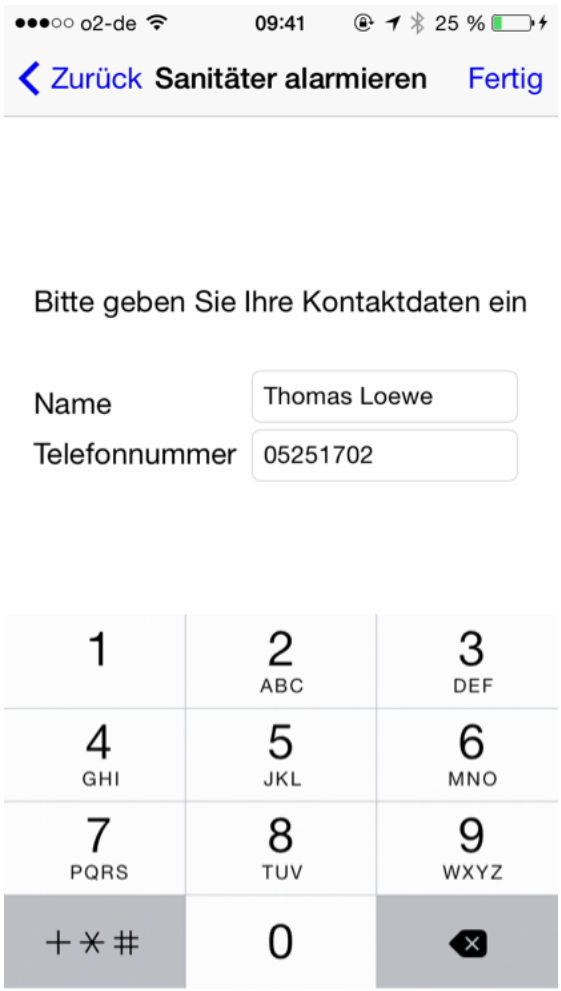


Abbildung 49: Eingabe der Kontaktdaten

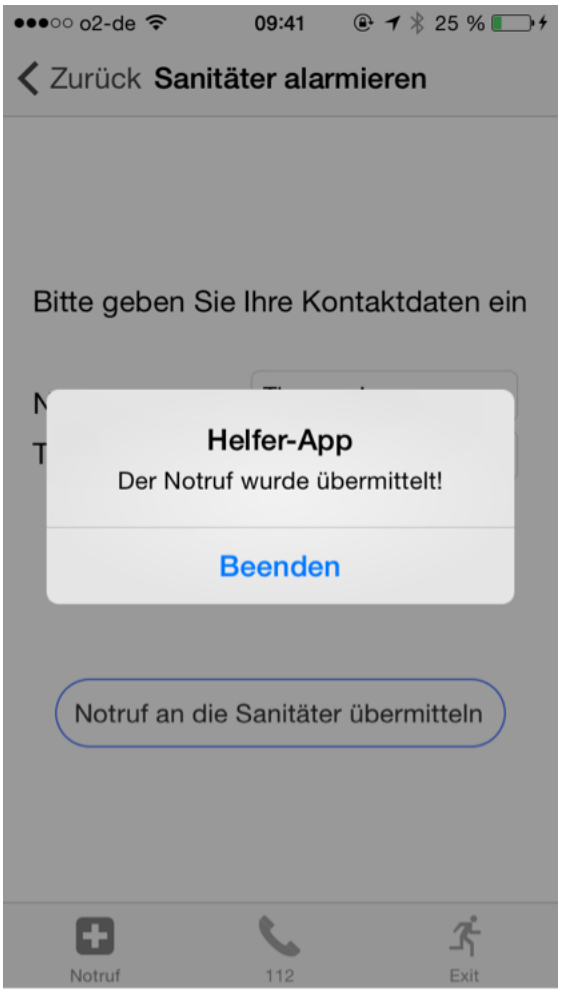


Abbildung 50: Der Notruf wurde übermittelt

5.5.1.2 Leitstelle anrufen

Über den Button *Notruf* hat der Ersthelfer die Möglichkeit, die Leitstelle über die 112 anzurufen. Die Abbildung 51 zeigt die Benutzeroberfläche.

Abbildung 51: Oberfläche 112 anrufen

5.5.1.3 Navigation zum Notausgang

Der Ersthelfer hat ebenfalls über die Helfer-App die Möglichkeit, sich zum nächsten Notausgang navigieren zu lassen. Über den Button *Exit* gelangt der Ersthelfer zu der Navigationsoberfläche. Ein Pfeil zeigt dabei, abhängig von der Ausrichtung des iPhones nach Norden, in die Richtung des nächsten Notausgangs.+491751763899

Abbildung 52: Navigation zum Notausgang

5.5.1.4 Bestätigung des Notrufs

Sobald der Notruf durch die Sanitäter bestätigt wurde, erhält der Ersthelfer eine Nachricht. Zudem wird dem App-Symbol eine 1 hinzugefügt.

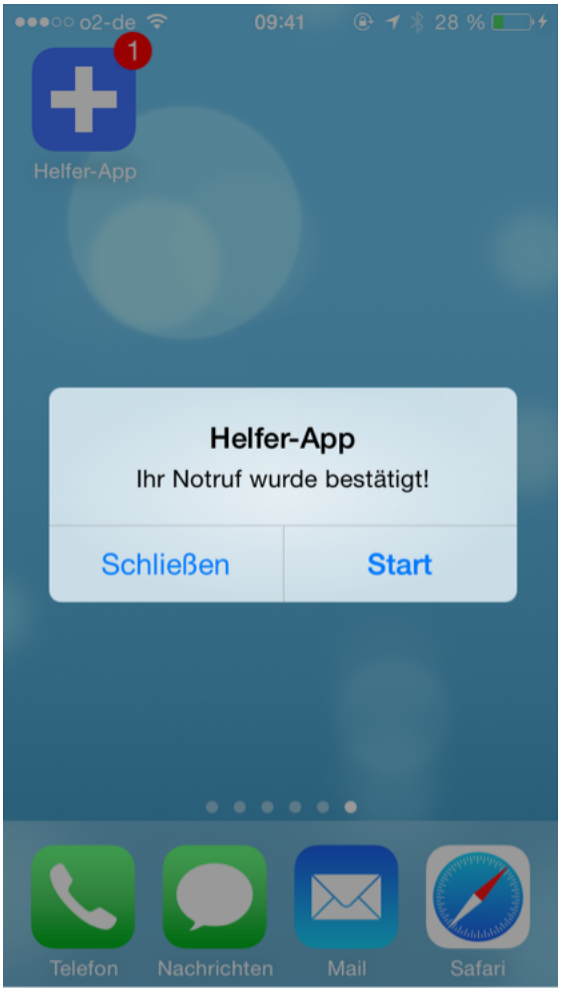


Abbildung 53: Der Notruf wurde bestätigt

5.5.2 Oberfläche der Sanitäter-App

Die Abbildung 54 zeigt die installierte Sanitäter-App auf dem iPad.

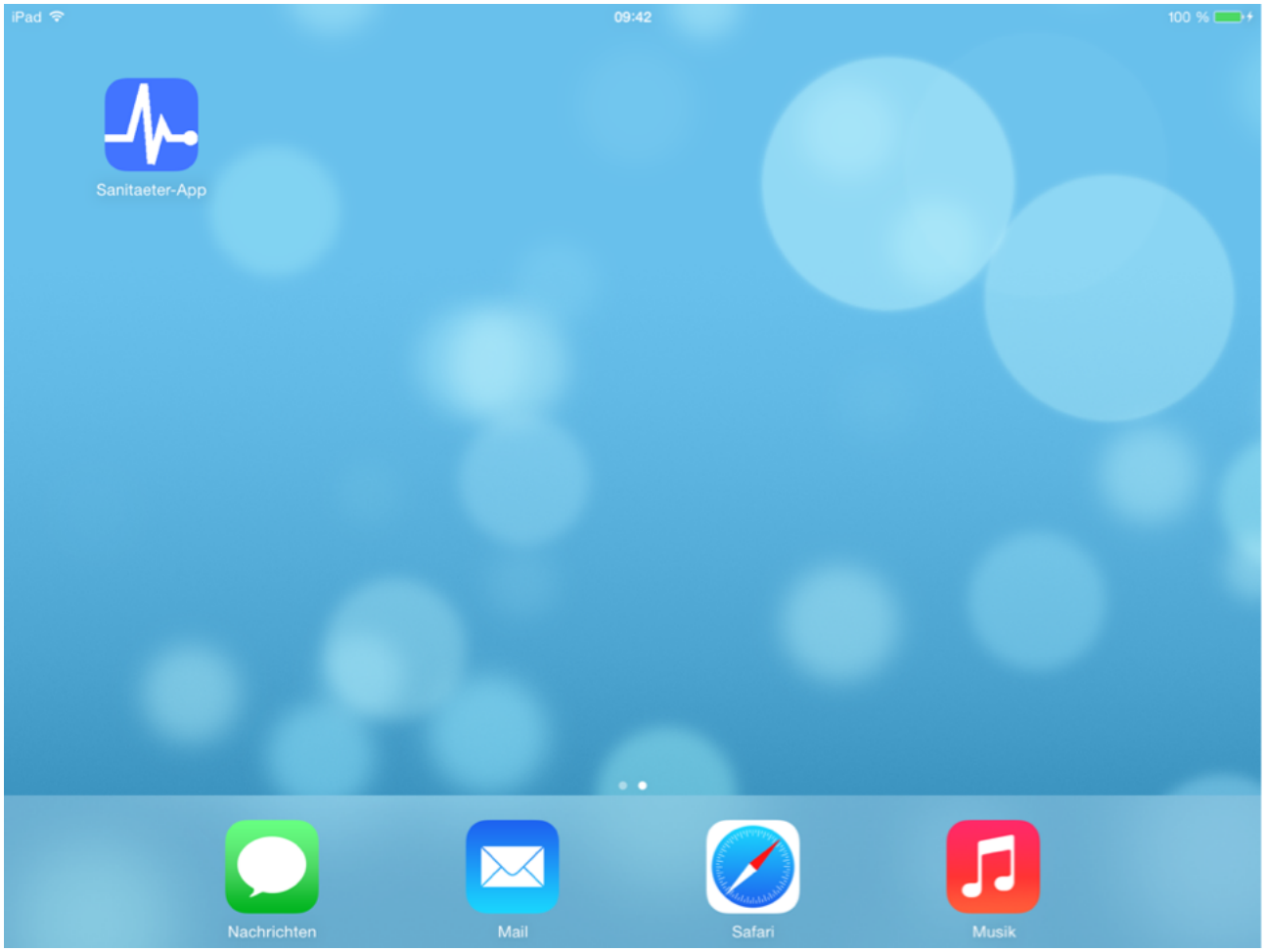


Abbildung 54: Installierte Sanitäter-App

5.5.2.1 Eintreffen eines neuen Notrufs

Trifft ein neuer Notruf ein, erhalten die Sanitäter eine Push Notification. Diese wird sowohl bei einem aktiven, wie auch bei einem gesperrtem iPad angezeigt.
Um eine schnelle Übersicht über alle nicht bestätigten Notrufe zu erhalten, wird die Anzahl der nicht bestätigten Notrufe dem App-Symbol hinzugefügt.

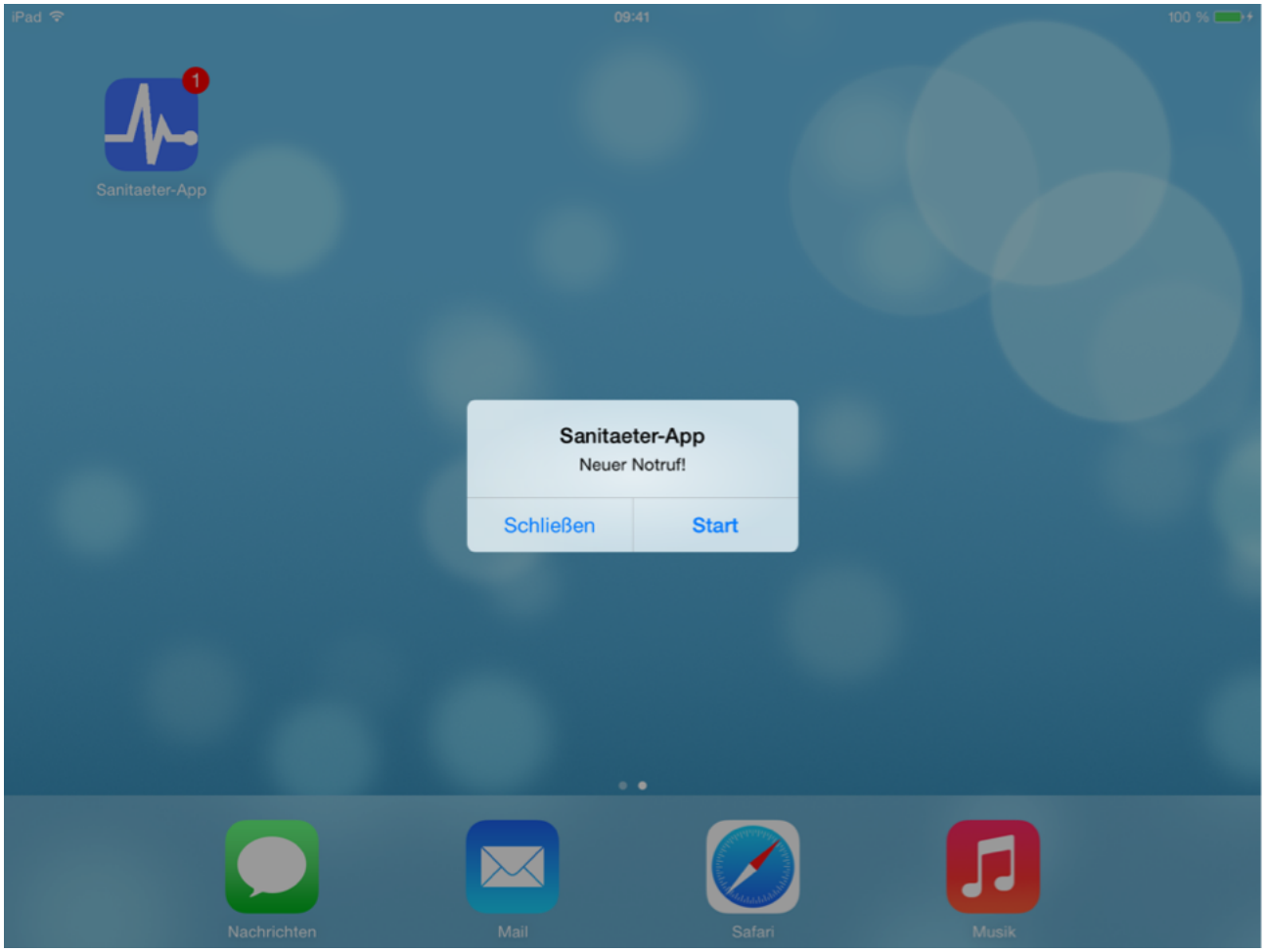


Abbildung 55: Neuer Notruf

5.5.2.2 Anzeigen der Notrufliste und der Patienteninformationen

Über den Button *Notrufliste* haben die Sanitäter die Möglichkeit, die Notrufinformationen abzurufen. Links befindet sich eine Liste mit allen Notrufen. Nicht bestätigte Notrufe werden in blauer Schriftfarbe angezeigt, bestätigte Notrufe erscheinen in schwarz. Rechts erhalten die Sanitäter weitere Informationen über den Notruf. Zudem haben die Sanitäter hier die Möglichkeit, zu jedem Patienten Notizen hinzuzufügen. Durch das Drücken auf den Button *2. Patient* wird den Sanitätern der zweite Patient angezeigt.

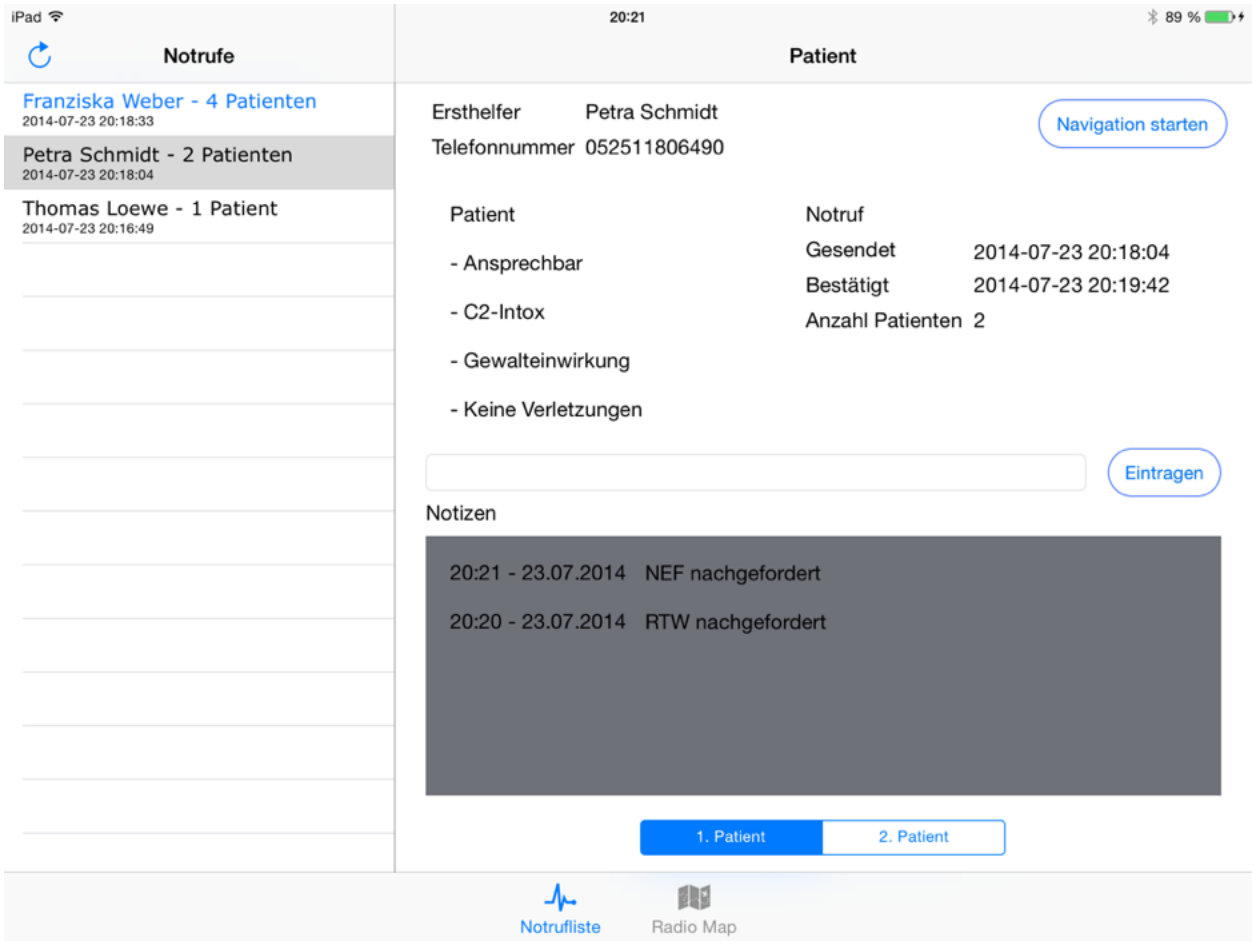


Abbildung 56: Notrufinformationen

5.5.2.3 Bestätigen eines Notrufs

Über den Button *Notruf bestätigen* haben die Sanitäter die Möglichkeit, einen Notruf zu bestätigen. Eine erfolgreiche Bestätigung wird den Sanitätern über einen Alert angezeigt. Zusätzlich färben sich der Name und die Patientenanzahl in der Notrufliste von blau zu schwarz. Des Weiteren wird aus dem *Notruf bestätigen*-Button ein *Navigation starten*-Button.

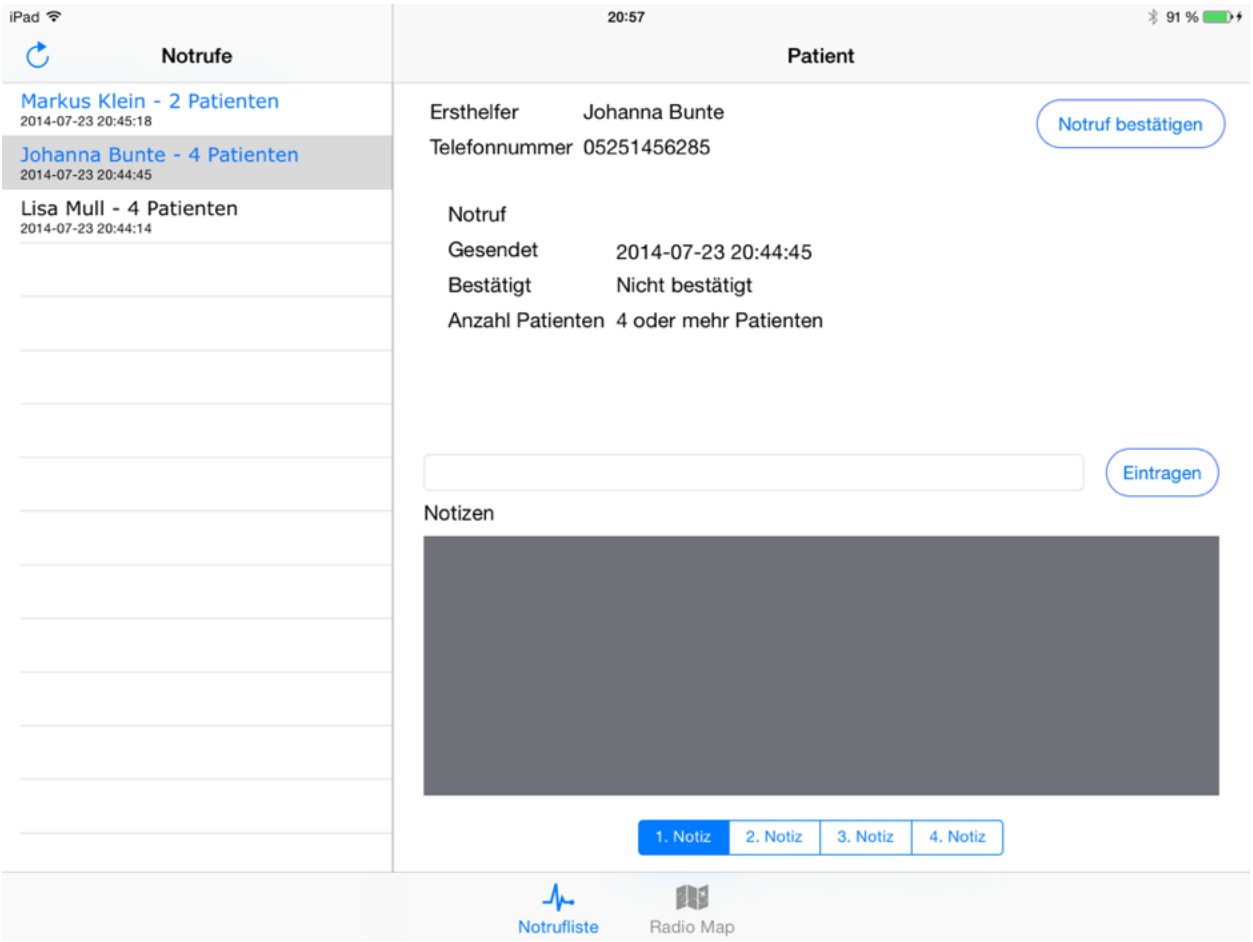
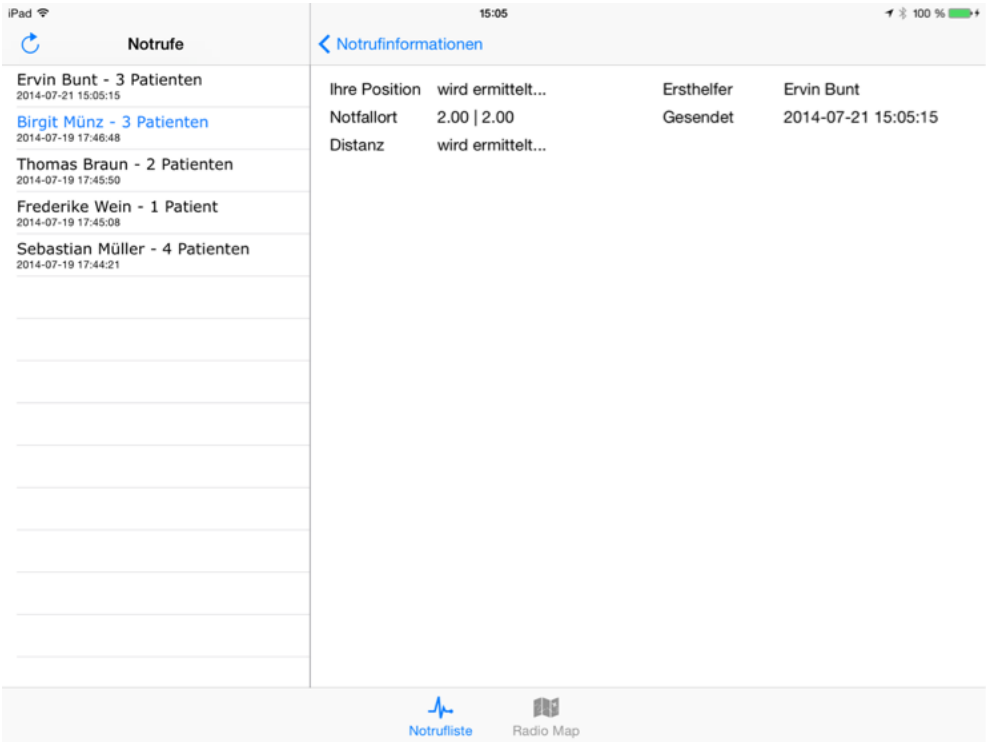


Abbildung 57: Notruf bestätigen

5.5.2.4 Navigation zum Notfallort

Wurde ein Notruf bestätigt, kann die Navigation zum Notfallort gestartet werden. Die Navigation erfolgt durch einen Pfeil, der den Sanitätern die Richtung zum Notfallort weist.



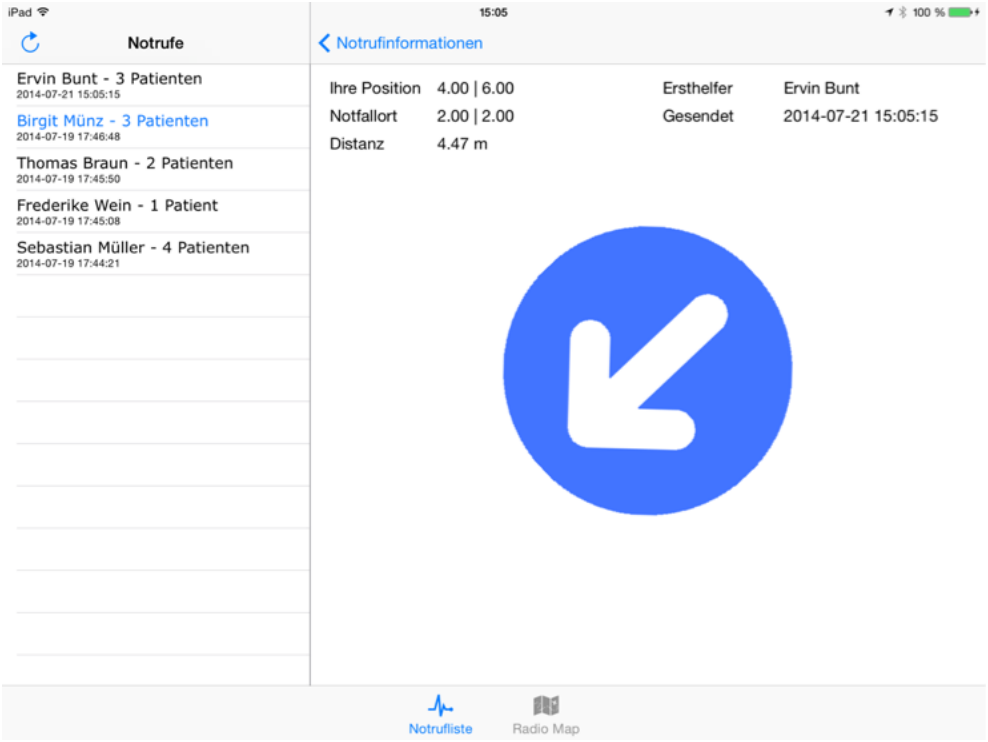


Abbildung 58 und 59: Navigation zum Notfallort

5.5.2.5 Erstellen der Radio Map

Als Voraussetzung für eine erfolgreiche Positionsbestimmung, müssen die Sanitäter eine Radio Map erstellen. Dies können sie unter dem Menüpunkt *Radio Map* durchführen. Beim Drücken des Buttons *Start* werden die Abstände vom iPad zu den Beacons gemessen. Der ermittelte Median wird anschließend der Radio Map hinzugefügt. Die Abbildung 60 zeigt die Mediane der Radio Map für die einzelnen Positionen.

Abbildung 60: Erstellen der Radio Map

6 Implementierung

6.1 Kommunikation mit den Estimote Beacons

Die Kommunikation mit den Estimote Beacons erfolgt über die Estimote SDK 2.0.0. Zur Verwaltung der Beacons dient die Klasse *ESTBeaconManager* der Estimote SDK. Zusätzlich wird ein Objekt der Klasse *ESTBeaconRegion* ebenfalls aus der Estimote SDK benötigt. Mit Hilfe dieser EST*BeaconRegion* wird angegeben nach welchen Beacons der *ESTBeaconManager* suchen soll. Dazu muss mindestens die UUID der Beacons angegeben werden. Die für die Helfer- und Sanitäter-App verwendete UUID ist eindeutig und wird nur für diese beiden Apps verwendet. Die Suche nach Beacons kann zusätzlich zur UUID auf einen minor- oder major-Wert beschränkt werden. Folgender Quellcode zeigt die Erstellung der UUID und die Zuweisung der *ESTBeaconRegion* zum *ESTBeaconManager* :

//Objekt der Klasse NSUUID mit der Beacon UUID wird erstellt.
*NSUUID *estimoteUUID = [[NSUUID alloc]*
initWithUUIDString:@„B56136C2-F66B-447C-83F2-C5F277AF6CE0"];

// Objekt der Klasse *ESTBeaconRegion* wird erstellt. Es wird nach allen *major- und minor-Werten* gesucht.
self.beaconRegion = [[ESTBeaconRegion alloc]
initWithProximityUUID:estimoteUUID identifier:@"RegionIdentifier"];

Folgender Quellcode grenzt die Suche nach Beacons mit der estimoteUUID und dem major-Wert 1 ein:

self.beaconRegion = [[ESTBeaconRegion alloc]
initWithProximityUUID:estimoteUUID major:1 minor:[self.beacon.minor
unsignedIntValue] identifier:@"RegionIdentifier"];

Anschließend wird dem *ESTBeaconManager* der Auftrag gegeben, nach den Beacons in der Region zu suchen:

// Objekt der Klasse ESTBeaconManager sucht nach Beacons in der Region
[self.beaconManager startRangingBeaconsInRegion:self.beaconRegion];

Wurde mindestens ein Beacon gefunden, wird folgende Methode aufgerufen:
*-(void)beaconManager:(ESTBeaconManager *)manager didRangeBeacons: (NSArray *)beacons inRegion:(ESTBeaconRegion *)region*

Diese Methode generiert das NSArray *beacons* mit allen Beacons in Reichweite des Endgerätes und der definierten Region. Das NSArray *beacons* ist nach den gemessenen Distanzen zwischen dem Endgerät und den Beacons sortiert. Das nächste Beacon zum Endgerät ist somit das erste Objekt des NSArray. Die minor- und major-Werte haben keinen Einfluss auf die Sortierung. Das nächste Beacon kann mit folgendem Quellcode aus dem NSArray *beacons* gelesen werden:

*ESTBeacon *firstBeacon =[beacons firstObject];*

Der Parameter *distance* erhält die gemessene Entfernung vom Beacon zum Endgerät in Form eines float-Wertes:

firstBeacon.distance;

Um den Mittelwert und den Median des Beacon 1 zu errechnen, werden die vorher gemessenen Distanzen des Beacon 1 in dem NSArray *werteBeacon1* gespeichert. Das Beacon 1 hat den minor-Wert 1 und kann bei drei Beacons in Reichweite des Endgerätes wie folgt ermittelt werden:

```
if([firstBeacon.minor isEqualToNumber:@1])
    [werteBeacon1 addObject:firstBeacon.distance];
else
if([secondBeacon.minor isEqualToNumber:@1])
    [werteBeacon1 addObject:secondBeacon.distance];
else
if([thirdBeacon.minor isEqualToNumber:@1])
    [werteBeacon1 addObject:thirdBeacon.distance];
```

Die if-Anweisungen sind erforderlich, da nicht bekannt ist an welcher Position sich das Beacon 1 im NSArray *beacons* befindet.

6.2 Bestimmung des Medians der Distanzen

Für die Positionsbestimmung in der Helfer- und Sanitäter-App wird nicht nur eine gemessene Distanz zwischen Endgerät und Beacons verwendet, sondern der Median aus mindestens drei bzw. zehn gemessenen Distanzen. In der Klasse *ExitNavigationViewController* der Helfer-App und in der Klasse *NavigationViewController* der Sanitäter-App, werden mindestens vier Werte zur Berechnung des Medians ermittelt.
Aus mindestens zehn gemessenen Werten wird dagegen der Median in der Klasse *AnzahlVerletzte* der Helfer-App und der Klasse *RadioMapViewController* der Sanitäter-App ermittelt
In allen Klassen gibt die Variable *wiederholungen : int* dabei die Mindestanzahl der gemessen Distanzen an, die für die Berechnung des Medians genutzt werden.
Sobald von allen drei Beacons die Mindestanzahl von Distanzen ermittelt wurde, werden keine weiteren Distanzen mehr hinzugefügt. Es ist jedoch möglich, dass z.B. durch das Beacon 1 fünf Signale beim Endgerät eintreffen, während erst drei Signale durch das Beacon 2 eingetroffen sind. Dadurch wird sichergestellt, dass eine Mindestanzahl von Messungen vorhanden ist und die Zeit für die Positionsbestimmung nicht zu lang dauert. Dennoch werden alle gemessenen Werte verwendet, um die Distanz zu optimieren.

Folgender Code zeigt, die Ermittlung des Medians:

```
if([werteBeacon1 count]>=wiederholungen)
  {
    [werteBeacon1 sortUsingSelector:@selector(compare:)];
     if([werteBeacon1 count] % 2 == 0)
        median1 =  [NSNumber numberWithFloat:([[werteBeacon1
objectAtIndex:([werteBeacon1 count]/2)-1] floatValue] + [[werteBeacon1
objectAtIndex:(([werteBeacon1 count]/2))] floatValue])/2];
     else
        median1 = [NSNumber numberWithFloat:[[werteBeacon1
objectAtIndex:([werteBeacon1 count]/2)] floatValue]];
     fertig1=YES;
  }
```

Zur Ermittlung des Medians wird das NSArray *beacons*, mit allen gemessen Distanzen zu einem Beacon, sortiert. Bei einer geraden Anzahl von gemessenen Werten, werden anschließend die beiden mittleren Werte addiert und durch zwei geteilt. Für vier gemessene Distanzen wird demnach der zweite und dritte Wert addiert.
Um letztendlich den Median zu erhalten, wird das Ergebnis durch zwei geteilt.
Eine ungerade Anzahl von gemessenen Werten, zur Ermittlung des Medians, wird weder in der Helfer-App, noch der Sanitäter verwendet. Dennoch ist eine ungerade Anzahl von Werten in der Implementierung berücksichtigt. Um den Median zu erhalten, wird der mittlere Wert des NSArrays *beacons* verwendet.
Bei einer Anzahl von drei gemessenen Werten, ist dies der zweite Wert des NSArrays *beacons*.

Das Vorgehen zur Ermittlung des Medians wird für die Beacons 2 und 3 wiederholt.

6.3 Positionsbestimmung des Endgerätes mittels Lateration

Eine Möglichkeit zur Positionsbestimmung ist das im Kapitel 2.4.1 beschriebene Verfahren der Lateration. Allerdings ergeben die Versuche aus Kapitel 4, dass das Fingerprinting-Verfahren deutlich bessere Ergebnisse liefert, als das Verfahren der Lateration. Aus diesem Grund wird in beiden Apps das Verfahren der Lateration nicht verwendet. Es wird lediglich für die Versuche aus Kapitel 4.3.1 benötigt.
Die Idee bei der Lateration ist es, den gemeinsamen Schnittpunkt, aller durch die Beacons gebildeten Kreise, zu errechnen. Dieser Schnittpunkt lässt sich durch das Lösen des folgenden linearen Gleichungssystems bestimmen.

$$(a)\ (a.x - x)^2 + (a.y - y)^2 = dA^2$$
$$(b)\ (b.x - x)^2 + (b.y - y)^2 = dB^2$$
$$(c)\ (c.x - x)^2 + (c.y - y)^2 = d\mathrm{C}^2$$

Gegeben seien dabei die Positionen der iBeacons A(a.x, a.y), B(b.x, b.y), C(c.x, c.y), sowie die Distanzen der einzelnen iBeacons zum Endgerät dA, dB und dC. Im realen Fall ist jedoch davon auszugehen, dass die Kreise keinen gemeinsamen Schnittpunkt haben und somit das lineare Gleichungssystem über keine eindeutige Lösung verfügt. Dies liegt an dem ungenauen Radius, also dem Abstand vom iBeacon zum Endgerät. Vielmehr ist davon auszugehen, dass sich die Kreise dem Endgerät nur nähern. Dabei ist zu beachten, ob die Kreise nicht bis zur Position des Endgerätes reichen, also keine Schnittpunkte bilden oder sie über die Position des Endgerätes reichen, also Schnittpunkte bilden. Bilden die Kreise keine Schnittpunkte, kann das reine Verfahren der Lateration nicht angewandt werden. Die Kreise müssen solange vergrößert werden, bis sich Schnittpunkte bilden. In dieser Implementierung wird nur der Fall betrachtet, indem die Kreise Schnittpunkte bilden.
Die Abbildung 61 beschreibt das Beispiel, indem sich bei dem Verfahren der Lateration die Kreise schneiden und insgesamt sechs Schnittpunkte bilden.

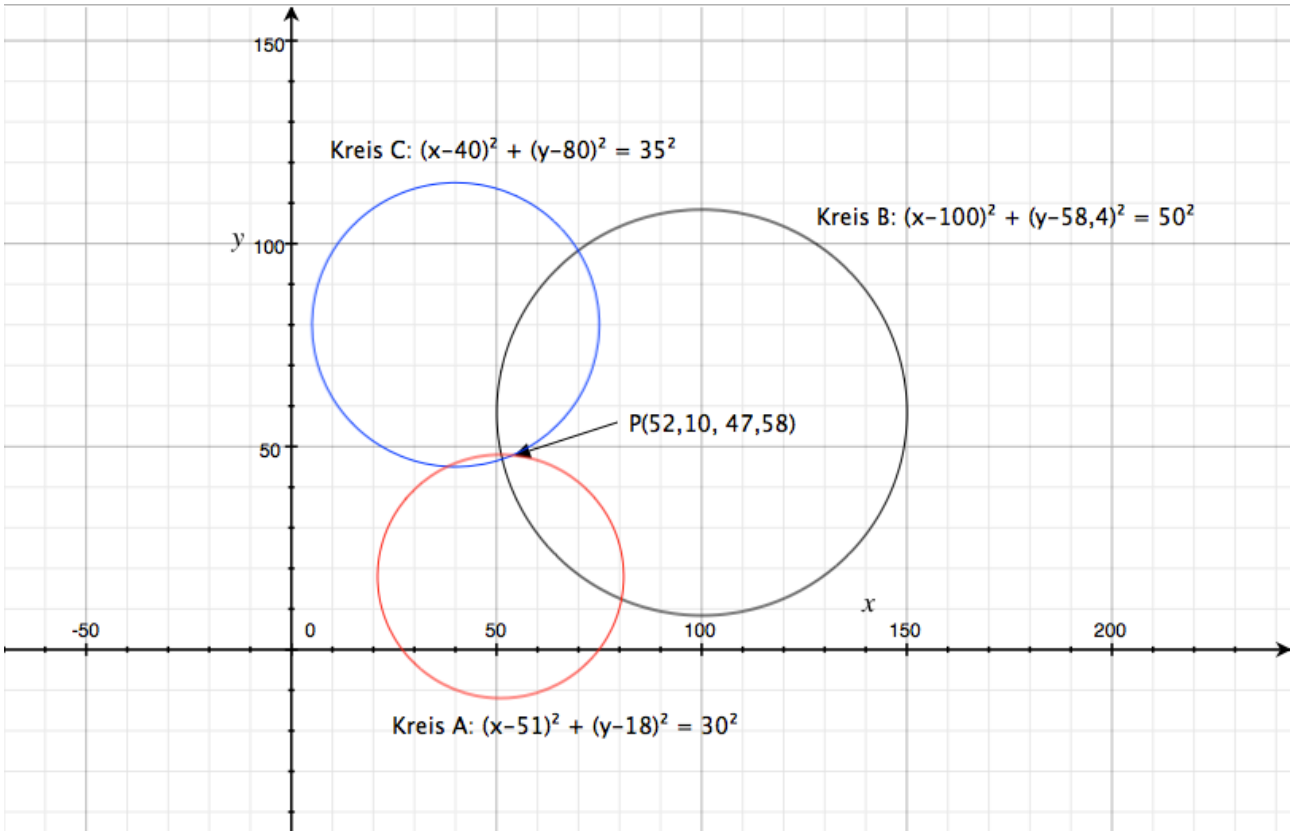


Abbildung 61: Näherung der Positionsbestimmung

Der Punkt P, in diesem Beispiel P(52,10 l 47,58), wird dabei durch die Methode *-(CGPoint)getPositionWithDistance:(float)distance1 Beacon1:(CGPoint)beacon1 Distance2:(float)distance2 Beacon2:(CGPoint)beacon2 Distance3:(float)distance3 Beacon3:(CGPoint)beacon3* der Klasse *LaterationManager* errechnet.

Die float-Werte distance1, distance2 und distance3 sind dabei die ermittelten Abstände von den Beacons 1, 2 und 3 zu dem Endgerät. In diesem Beispiel sind das 30, 50 und 35 m. Die CGPoints beacon1, beacon2 und beacon3 sind dabei die Positionen der Beacons im Koordinatensystem. Im diesem Beispiel sind das (51 l 18), (100 l 58,4) und (40 l 80).

Wie oben beschrieben ist davon auszugehen, dass das lineare Gleichungssystem mit drei Kreisgleichungen kein Ergebnis liefert, da es keinen genauen Schnittpunkt gibt. Theoretisch können zwei Kreise keinen, zwei oder unendlich viele Schnittpunkte bilden. Letzteres trifft nur auf Kreise zu, die identisch sind, die beiden Beacons müssten sich dafür an derselben Position befinden. Auf dieses Beispiel wird nicht näher eingegangen, da die Beacons für die Positionsbestimmung im Raum verteilt werden.

Zur Ermittlung von zwei Schnittpunkten wird die Methode -*(NSMutableArray *)getInterceptWithBeacon1:(CGPoint)b1 Beacon2:(CGPoint)b2 Distance1:(CGFloat)d1 Distance2:(CGFloat)d2* der Klasse *LaterationManager* verwendet. Im Beispiel der Abbildung 61 werden die zwei Schnittpunkte durch Subtraktion der Kreisgleichung von Kreis A und von Kreis B ermittelt. Als Ergebnis der Subtraktion steht zuerst die Chordale, also eine Gerade die durch beide Schnittpunkte verläuft. Für die Kreise A und B ist die Chordale in Abbildung 62 zu sehen.

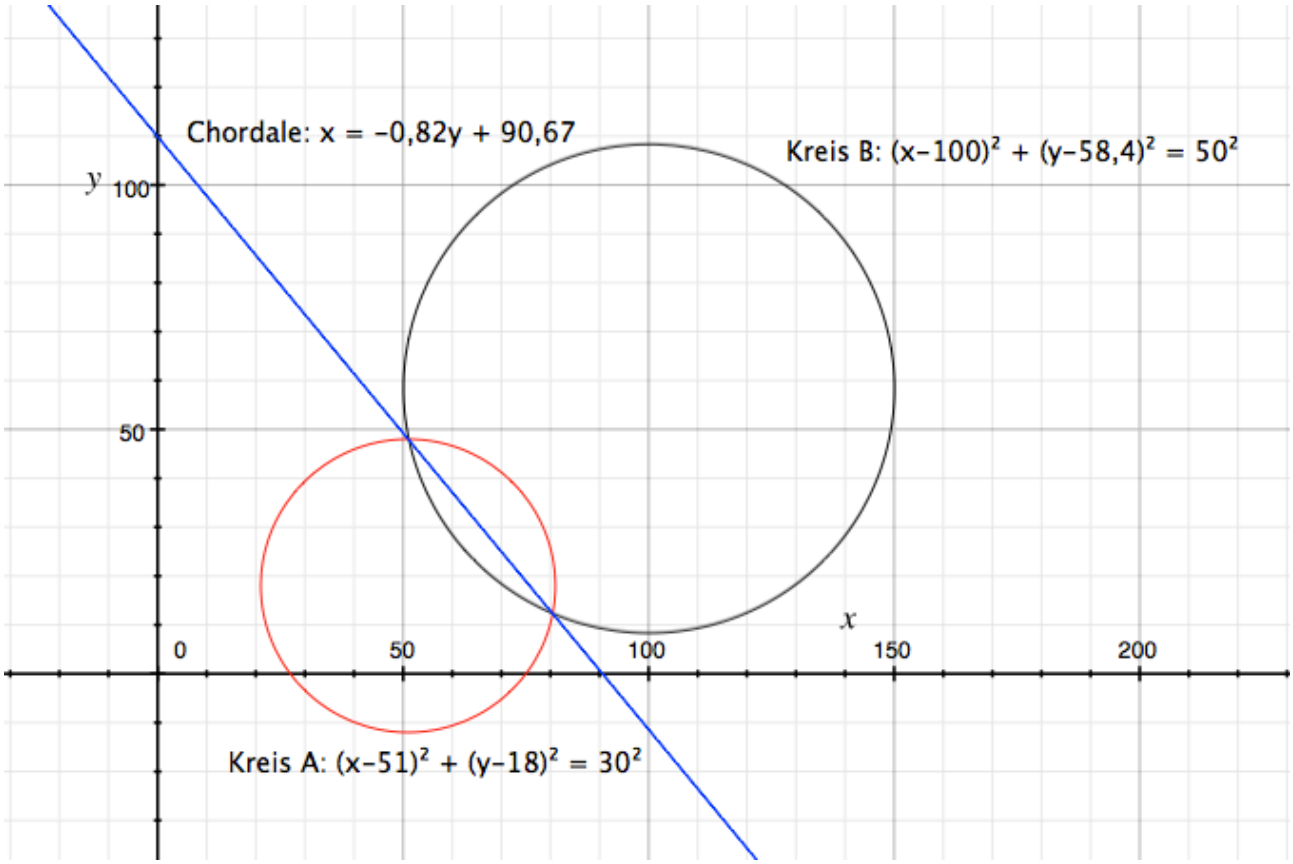


Abbildung 62: Chordale der Kreise A und B

Folgender Code errechnet die Chordale zweier Kreise:

//xWert. Durch xWert wird der yWert geteilt, damit auf der linken Seite nur ein x steht.
CGFloat xWert = 2(-b1.x+b2.x);*

// y ausrechnen
// Um wirklich ein ganzes x auf einer Seite zu haben, teilen wir hier durch -x
double yWert =(2(-b1.y+b2.y)) / -xWert;*

*double restWert = (((b1.x*b1.x) + (b1.y * b1.y) - (d1 * d1)) - ((b2.x*b2.x) + (b2.y * b2.y) - (d2 * d2)))/-xWert;*

Für die Chordale ergibt sich nun die Gleichung: x = yWert * y + restWert

Um die Schnittpunkte der Kreise zu erhalten, wird die Chordale in die Kreisgleichung A eingesetzt. Durch das Anwenden der p-q-Formel erhält man beide y-Werte der Schnittpunkte. Dies geschieht im folgenden Quellcode:

```
// Die Chordale wird nun in die erste Kreisgleichung eingesetzt

//Binomische Formel ausrechnen.
double y2WertAB = (yWert * yWert) +1;
        if(y2WertAB<0)
       y2WertAB = y2WertAB *-1;

//Hier und im nächsten Schritt durch y2WertAB teilen, damit die Voraussetzungen für die PQ-Formel gegeben sind

double yWertAB = ((2*yWert*(-b1.x+restWert))+ (-2*b1.y)) / y2WertAB ;

double restWertAB = ((-b1.x+restWert) *(-b1.x+restWert) + (b1.y * b1.y) - (d1 * d1))/ y2WertAB;

//Jetzt mit p-q-Formel auflösen. Wir erhalten die Y-Werte der beiden Schnittpunkte

    a.y = -yWertAB/2 + sqrt(((yWertAB/2)*(yWertAB/2)-restWertAB));
    b.y = -yWertAB/2 - sqrt(((yWertAB/2)*(yWertAB/2)-restWertAB));
```

Die dazugehörigen x-Werte werden durch Einsetzen der y-Werte in die Chordale ermittelt.

```
a.x =  yWert * a.y + restWert;
b.x = yWert * b.y + restWert;
```

Als Ergebnis liefert die Methode *-(NSMutableArray *)getInterceptWithBeacon1:(CGPoint)b1 Beacon2:(CGPoint)b2 Distance1:(CGFloat)d1 Distance2:(CGFloat)d2* ein *NSMutableArray* mit den beiden Schnittpunkten. Die Methode wird insgesamt dreimal von der Methode -

(CGPoint)getPositionWithDistance:(float)distance1 Beacon1:(CGPoint)beacon1 Distance2:(float)distance2 Beacon2:(CGPoint)beacon2 Distance3:(float)distance3 Beacon3:(CGPoint)beacon3 ausgerufen. Dadurch werden insgesamt sechs Schnittpunkte errechnet. Aus diesen Schnittpunkten werden wiederum acht Dreiecke gebildet. Das Dreieck mit dem kleinsten Umfang befindet sich am nächsten zum Endgerät.

Die Ermittlung der acht Dreiecke erfolgt im folgenden Quellcode:

```
// erster Schnittpunkt von Kreis A und B
CGPoint a1 = [pointA[0] CGPointValue];

// zweiter Schnittpunkt von Kreis A und B
CGPoint a2 = [pointA[1] CGPointValue];

// erster Schnittpunkt von Kreis A und C
CGPoint b1 = [pointB[0] CGPointValue];

// zweiter Schnittpunkt von Kreis A und C
CGPoint b2 = [pointB[1] CGPointValue];

// erster Schnittpunkt von Kreis B und C
CGPoint c1 = [pointC[0] CGPointValue];

// zweiter Schnittpunkt von Kreis B und C
CGPoint c2 = [pointC[1] CGPointValue];

// Es werden nun die drei Punkte bestimmt, die das kleinste Dreieck bilden, dazu werden mit Hilfe der euklidischen Distanz alle Strecken ausgerechnet A1B1, B1C1, A1B2 usw...

 //Abstand erster Schnittpunkt vom Kreis A und B zum ersten Schnittpunkt vom Kreis A und C
    double A1B1 = sqrt(pow (a1.x - b1.x, 2) + pow (a1.y - b1.y, 2));

//Abstand erster Schnittpunkt vom Kreis A und C zum ersten Schnittpunkt vom Kreis B und C
    double B1C1 = sqrt(pow (b1.x - c1.x, 2) + pow (b1.y - c1.y, 2));

//Abstand erster Schnittpunkt vom Kreis A und B zum zweiten Schnittpunkt vom Kreis A und C
    double A1B2 = sqrt(pow (a1.x - b2.x, 2) + pow (a1.y - b2.y, 2));
```

//Abstand zweiter Schnittpunkt vom Kreis A und C zum ersten Schnittpunkt vom Kreis B und C

double B2C1 = sqrt(pow (b2.x - c1.x, 2) + pow (b2.y - c1.y, 2));

//Abstand erster Schnittpunkt vom Kreis A und C zum ersten Schnittpunkt vom Kreis B und C

double B1C2 = sqrt(pow (b1.x - c2.x, 2) + pow (b1.y - c2.y, 2));

//Abstand zweiter Schnittpunkt vom Kreis A und C zum zweiten Schnittpunkt vom Kreis B und C

double B2C2 = sqrt(pow (b2.x - c2.x, 2) + pow (b2.y - c2.y, 2));

//Abstand zweiter Schnittpunkt vom Kreis A und B zum ersten Schnittpunkt vom Kreis A und C

double A2B1 = sqrt(pow (a2.x - b1.x, 2) + pow (a2.y - b1.y, 2));

//Abstand zweiter Schnittpunkt vom Kreis A und B zum zweiten Schnittpunkt vom Kreis A und C

double A2B2 = sqrt(pow (a2.x - b2.x, 2) + pow (a2.y - b2.y, 2));

//Abstand erster Schnittpunkt vom Kreis A und B zum ersten Schnittpunkt vom Kreis B und C

double A1C1 = sqrt(pow (a1.x - c1.x, 2) + pow (a1.y - c1.y, 2));

//Abstand erster Schnittpunkt vom Kreis A und B zum zweiten Schnittpunkt vom Kreis B und C

double A1C2 = sqrt(pow (a1.x - c2.x, 2) + pow (a1.y - c2.y, 2));

//Abstand zweiter Schnittpunkt vom Kreis A und B zum zweiten Schnittpunkt vom Kreis B und C

double A2C2 = sqrt(pow (a2.x - c2.x, 2) + pow (a2.y - c2.y, 2));

//Abstand zweiter Schnittpunkt vom Kreis A und B zum ersten Schnittpunkt vom Kreis B und C

double A2C1 = sqrt(pow (a2.x - c1.x, 2) + pow (a2.y - c1.y, 2));

//Umfang der Dreiecke berechnen. Im Kleinsten liegt das Endgerät

```
double A1B1C1 = A1B1 + B1C1 + A1C1;
double A1B2C1 = A1B2 + B2C1 + A1C1;
double A1B1C2 = A1B1 + B1C2 + A1C2;
double A1B2C2 = A1B2 + B2C2 + A1C2;
double A2B1C1 = A2B1 + B1C1 + A2C1;
double A2B2C1 = A2B2 + B2C1 + A2C1;
double A2B1C2 = A2B1 + B1C2 + A2C2;
double A2B2C2 = A2B2 + B2C2 + A2C2;
```

Als Position des Endgerätes werden die beiden Mittelwerte des x- und y-Werts der drei Punkte gewählt. Dies wird in Abbildung 63 dargestellt und mit Hilfe des folgenden Quellcodes errechnet:

```
m.x = (bestPointAPoint.x + bestPointBPoint.x + bestPointCPoint.x) /3;
m.y = (bestPointAPoint.y + bestPointBPoint.y + bestPointCPoint.y) /3;
```

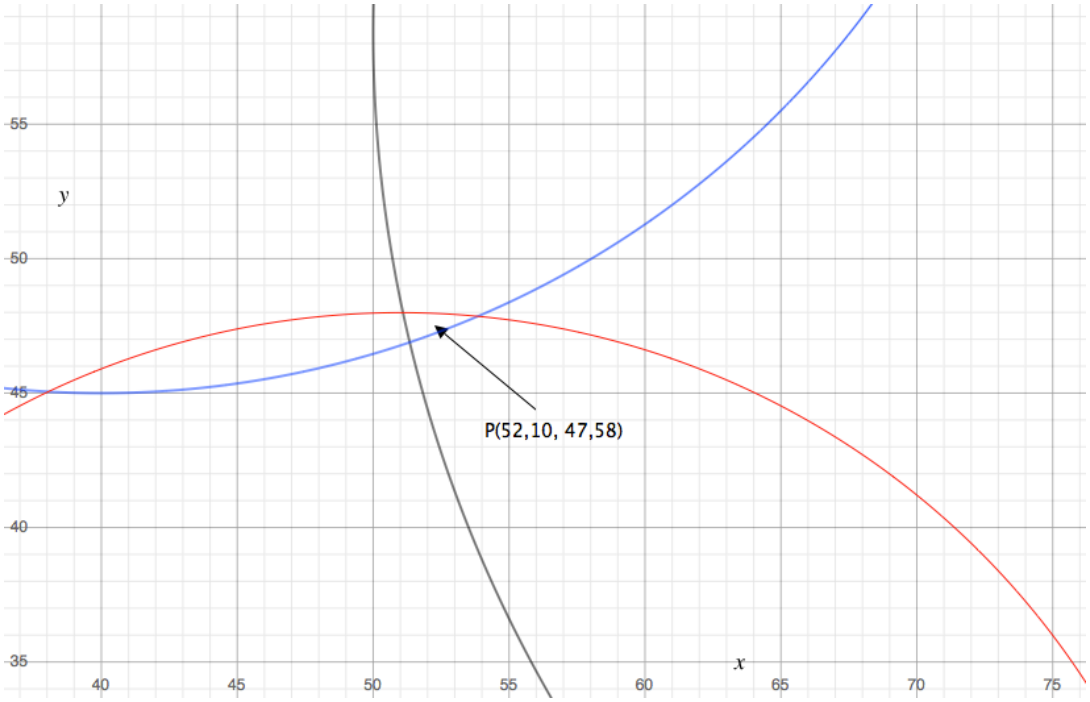


Abbildung 63: Errechnete Position des Endgerätes

6.4 Positionsbestimmung des Endgerätes mittels Fingerprinting

Die Versuche aus Kapitel 4 zeigen, dass das Fingerprinting-Verfahren genauere Ergebnisse als das Verfahren der Lateration liefern. Aus diesem Grund wird für beide Apps zur Positionsbestimmung und somit auch zur Navigation das Fingerprinting-Verfahren angewendet. Für die Positionsbestimmung wird zuvor eine Radio Map erstellt, welche von der Klasse *RadioMap* verwaltet wird. Das Vorgehen zur Erstellung der Radio Map wird in Kapitel 2.4.2 beschrieben. Die Methode *-(NSMutableArray *)getRadioMap* der Klasse *RadioMap* liefert die aktuelle Radio Map.

Die Ermittlung der Position des Endgerätes erfolgt in der Methode *-(CGPoint)getPosition:(CGFloat)dBeacon1 distance:(CGFloat)dBeacon2 distance3:(CGFloat)dBeacon3* der Klasse *FingerprintManager.*
Zur Positionsbestimmung werden zuerst die Distanzen von den Beacons zu dem Endgerät ermittelt. Dies geschieht nach dem Vorgehen aus Kapitel 6.1. Die ermittelten Distanzen werden anschließend an die Methode *-(CGPoint)getPosition:(CGFloat)dBeacon1 distance:(CGFloat)dBeacon2 distance3:(CGFloat)dBeacon3* der Klasse *FingerprintManager* übergeben.

In dieser Methode werden die Distanzen mit allen gespeicherten Distanzen der Radio Map verglichen. Als Ähnlichkeitsmaß dient die euklidische Distanz. Sie wird in der Methode *-(BOOL)comparePointsWithDistance: (CGFloat)d1 distance2:(CGFloat)d2 distance3:(CGFloat)d3 dRadioMap1: (CGFloat)b1 dRadioMap2:(CGFloat)b2 dRadioMap3:(CGFloat)b3 distance: (float)d* der Klasse *FingerprintManager* wie folgt errechnet:

float neueDistanz = sqrt((b1-d1)(b1-d1) + (b2-d2)*(b2-d2) + (b3-d3)*(b3-d3));*

Die Parameter *d1*, *d2* und *d3* sind dabei die gemessen Distanzen von den Beacons 1, 2 und 3 zum Endgerät. Die Parameter *b1*, *b2* und *b3* sind die Vergleichswerte aus der Radio Map. Ist der ermittelte Wert kleiner, als der zuvor kleinste ermittelte Wert, wird die Variable *neueDistanz* mit der neuen Distanz überschrieben. Anschließend liefert die Methode *true*. Die Methode *-(CGPoint)getPosition:(CGFloat)dBeacon1 distance:(CGFloat)dBeacon2 distance3:(CGFloat)dBeacon3* speichert darauf die Position des Endgeräts mit den Werten b1, b2, und b3.

6.5 Ausrichtung des Navigationspfeils

Nachdem der *FingerprintManager* die Position des Endgeräts bestimmt hat, wird der Navigationspfeil ausgerichtet. Dazu müssen die Variablen *CGPoint myPos, CGPoint targetPos* und *nordenAusrichtung* in der Klasse *PfeilImageView* überschrieben werden. Diese Einstellungen werden in der jeweiligen ViewController-Klasse vorgenommen. Bei der Sanitäter-App ist dies die *NavigationViewController*-Klasse und bei der Helfer-App die *ExitNavigationViewController*-Klasse.

```
//Position des Notfallorts in der Sanitäter-App
targetPos = self.notruf.pos;

//Position des nächsten Notausganges in der Helfer-App
CGPoint targetPos = [self.finerprintManager getEmergencyExit:pos];

//Ausrichtung der Radio Map nach Norden
[self.pfeil
setNordenAusrichtung:self.finerprintManager.radioMap.nordenAusrichtung]
;

//Setzen der Start und Ziel Positionen
[self.pfeil setMyPos:pos];
[self.pfeil setTargetPos:targetPos];
```

Sobald sich die Ausrichtung des Endgeräts nach Norden ändert, wird der Pfeil durch die Methode

```
-(void)locationManager:(CLLocationManager *)manager
didUpdateHeading:(CLHeading *)newHeading
```

der Klasse *PfeilImageView* neu ausgerichtet. Sollte sich das Endgerät nicht bewegen, kann die Ausrichtung manuell durch den Aufruf *[self.pfeil update]* gestartet werden.

Die Methode Update der Klasse *PfeilImageView* bewirkt durch den Aufruf

```
[self.locationManager startUpdatingLocation];
[self.locationManager startUpdatingHeading];
```

ebenfalls die Ausführung der Methode

```
-(void)locationManager:(CLLocationManager *)manager
didUpdateHeading:(CLHeading *)newHeading.
```

In dieser Methode wird zur Justierung des Pfeils der Steigungswinkel der Geraden, die den Start- und den Zielpunkt verbindet, berechnet. Dies ist nur erforderlich, wenn die x- und y-Werte des Start- und Zielpunktes verschieden sind. Sind dagegen die x- oder y-Werte gleich, zeigt der Pfeil in eine der vier Himmelsrichtungen. Sind die x- und die y-Werte gleich, ist das Ziel erreicht.

Der folgende Quellcode zeigt die Berechnung des Steigungswinkels:

```
if((self.myPos.x!=self.targetPos.x) && (self.myPos.y!=self.targetPos.y))
 {
   winkel= (self.myPos.y-self.targetPos.y)/(self.myPos.x-self.targetPos.x);

   //Arkustangens
    winkel = atanf(winkel);

    //Anschließend wird der Radiant in Grad umgewandelt:

    //M_PI steht für die Zahl PI.
  #define RADIANS_TO_DEGREES(radians) ((radians) * (180.0 / M_PI))
   winkel = RADIANS_TO_DEGREES(winkel);
 }
```

Der jeweilige x- und y-Wert des Startpunktes kann größer, kleiner oder gleich des x- bzw. y-Werts des Zielpunkts sein. Dadurch kann die Gerade zwischen dem Start- und Zielpunkt eine negative, eine positive oder gar keine Steigung haben. Aus diesem Grund und aus dem Grund, dass der Steigungswinkel auf die Gradzahl des Kompasses aufgerechnet werden soll, muss zwischen den verschiedenen Start- und Zielkombinationen unterschieden werden.

Dies geschieht im folgenden Quellcode:

```
 if((self.myPos.x>self.targetPos.x) && (self.myPos.y==self.targetPos.y))
   winkel =-90;
   else
if((self.myPos.x>self.targetPos.x) && (self.myPos.y>self.targetPos.y))
   winkel = -90 -winkel;
   else
if((self.myPos.x<self.targetPos.x) && (self.myPos.y==self.targetPos.y))
   winkel =90;
   else
if((self.myPos.x<self.targetPos.x) && (self.myPos.y>self.targetPos.y))
   winkel = 90-winkel;
   else
if((self.myPos.x<self.targetPos.x) && (self.myPos.y<self.targetPos.y))
   winkel = 90-winkel;
   else
if((self.myPos.x==self.targetPos.x) && (self.myPos.y==self.targetPos.y))
   winkel =0;
   else
if((self.myPos.x==self.targetPos.x) && (self.myPos.y>self.targetPos.y))
   winkel = 180;
   else
if((self.myPos.x==self.targetPos.x) && (self.myPos.y<self.targetPos.y))
   winkel =0;
```

Für *self.myPos.x>self.targetPos.x) && (self.myPos.y<self.targetPos.y)* muss der Winkel nicht verändert werden.

Als nächster Schritt werden der errechnete Winkel und die Ausrichtung der Radio Map nach Norden auf die Ausrichtung des Endgerätes nach Norden addiert. Dies geschieht einmal für die alte Ausrichtung nach Norden des Endgeräts und einmal für die neue Ausrichtung des Endgeräts nach Norden.

*float alteRichtung = (-manager.heading.trueHeading + self.nordenAusrichtung +winkel) * M_PI / 180.0f;*

*float neueRichtung = (-newHeading.trueHeading + self.nordenAusrichtung +winkel) * M_PI / 180.0f;*

Über die *CABasicAnimation* wird der Navigationspfeil anschließend von der alten Position zur neuen Position gedreht.

```
CABasicAnimation *basicAnimation =[CABasicAnimation
animationWithKeyPath:@„transform.rotation"];

basicAnimation.fromValue = [NSNumber numberWithFloat:alteRichtung];
basicAnimation.toValue=[NSNumber numberWithFloat:neueRichtung];

[self.layer addAnimation:basicAnimation forKey:@"ausrichten"];
self.transform = CGAffineTransformMakeRotation(neueRichtung);
```

Sobald sich das Endgerät dreht oder eine neue Position des Endgeräts ermittelt wird, wiederholt sich die Ausrichtung des Pfeils.

6.5 Bereitstellen der Notrufinformationen

Der Webserver stellt für die Sanitäter-App ein JSON-Objekt mit Notrufinformationen bereit. Dieses Objekt kann von jeder installierten Sanitäter-App aufgerufen und verarbeitet werden.

Die nötigen Notrufinformationen werden zuvor von der Helfer-App in einer MySQL Datenbank gespeichert.

Das JSON-Objekt ist nach folgendem Muster aufgebaut:

[{ Notruf:
[{ id, gesendet, bestaetigt, posX, posY, patientenAnzahl, name, telefonnumer},
{Patient:
[
{id, ansprechbar, atmung, alkohol, orierntierungslos, sturz, gewalt, paediatrie, notfallsituationSonstige, kreislauf, herzanfall, atemnot, kopfschmerzen, kranpfanfall, vergiftung, erkrankungKeine, erkrankungSonstige, blutung, fremdkoerper, verbrennung, extremitaeten, kopfverletzung, augen, verletzungKeine, verletzungSonstige}
,{evt. bis zu 3 weitere Patienten}
]
}]
}]

Der Notruf enthält dabei abhängig von der ausgewählten Patientenanzahl einen, zwei, drei oder vier Patienten, selbst wenn zu dem dritten und vierten Patienten keine Patienteninformationen hinterlegt sind. Dadurch kann eine Änderung bei der Speicherung der Patienteninformationen erfolgen, ohne die Eigenschaften des Webservers ändern zu müssen.

In der Methode *-(NSMutableArray *) getData* der Klasse *WebserverConnection* lädt die Sanitäter-App das JSON-Objekt vom Webserver und erstellt die dazugehörigen Objekte der Klassen *Notruf* und *Patient*.

Folgender Befehl zeigt beispielhaft wie der Name des Ersthelfers aus dem JSON-Objekt in einem Objekt der Klasse *Notruf* gespeichert wird.

[myNotruf setName:[[key valueForKeyPath:@"Notruf.name"] objectAtIndex: 0]];

Das Speichern der BOOL-Variable *Ansprechbar* bei einem Objekt der Klasse *Patient* erfolgt für den ersten Patienten wie folgt:

[[myNotruf.patientArray objectAtIndex:0] setAnsprechbar:[[[[key valueForKeyPath:@"Notruf.Patient.ansprechbar"] objectAtIndex:1] objectAtIndex:0] boolValue]];

Für den zweiten Patienten müssen die rot unterlegten Index-Werte angepasst werden:

[[myNotruf.patientArray objectAtIndex:1] setAnsprechbar:[[[[key valueForKeyPath:@"Notruf.Patient.ansprechbar"] objectAtIndex:1] objectAtIndex:1] boolValue]];

Dies wird für jede Variable und jeden Patienten durchgeführt. Schließlich wird der Notruf einem NSMutableArray zugefügt.

[NotrufArray addObject:myNotruf];

Dieses Array ist gleichzeitig der Rückgabewert der Methode
return NotrufArray;

7 Test der Applikationen

In diesem Kapitel werden folgende Punkte untersucht:

1) Werden die Notrufinformationen korrekt von der Helfer-App an die Sanitäter-App übermittelt? Werden diese in der Sanitäter-App korrekt angezeigt?

2) Ist die Benachrichtigung über einen neuen Notruf in der Sanitäter-App korrekt?

3) Funktioniert die Bestätigung eines Notrufes?

4) Wird der Standort durch die Helfer-App korrekt ermittelt und erfolgreich an die Sanitäter-App übertragen? Wie genau ist die Positionsbestimmung?

5) Wird der Navigationspfeil korrekt ausgerichtet?

6) Wird die Distanz zwischen dem Start- und Zielpunkt korrekt berechnet?

7) Ist die Navigation in der Sanitäter-App zum Patienten korrekt? Wie genau ist diese?

8) Wird der nächste Notausgang in der Helfer-App korrekt ermittelt?

9) Wie genau ist die Navigation zum nächsten Notausgang in der Helfer-App?

7.1 Übertragung der Notrufinformationen

Beschreibung:

In diesem Test wird überprüft, ob die Notrufinformationen korrekt an die Sanitäter-App übermittelt werden. Dazu werden verschiedene Notrufe mit einem, zwei und vier Patienten in unterschiedlichen Notfallsituationen mit verschiedenen Erkrankungen und Verletzungen an die Sanitäter übermittelt.

Test mit einem Patienten:

Durchführung:

Folgendes Szenario wird ausgewählt:

Patientanzahl	1
Datum und Zeit des Notrufs	23. Juli 2014 - 20:36 Uhr
Patient ansprechbar	Ja
Notfallsituation	Sonstige
Erkrankungen	Herzanfall
Verletzungen	Keine
Name des Ersthelfers	Manuel Sauer
Telefonnumer des Ersthelfers	05251456228

Ergebnis:

Die Abbildung 64 zeigt, dass die Notrufinformationen erfolgreich übermittelt wurden.

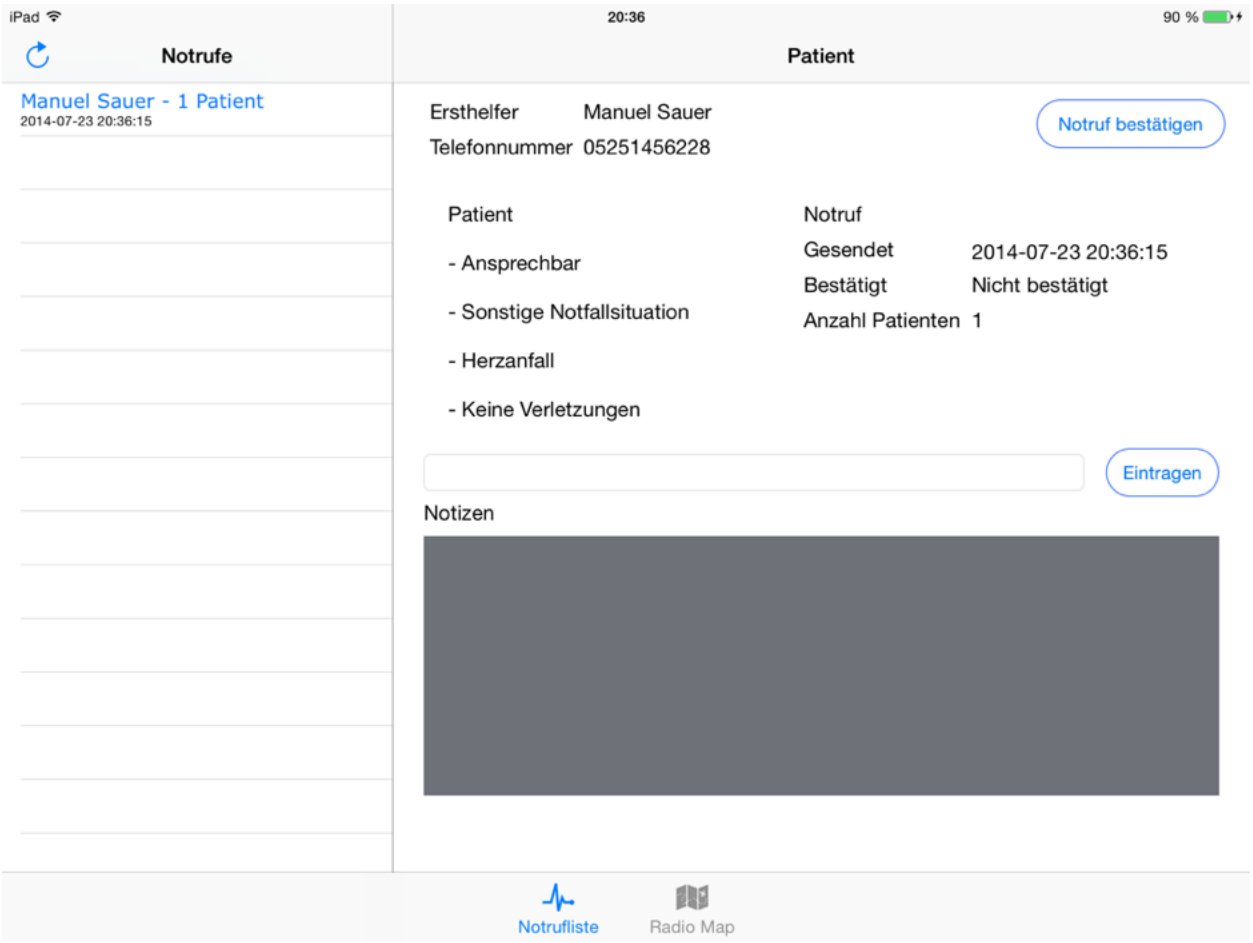


Abbildung 64: Übermittlung der Notrufinformationen

Test mit zwei Patienten:

Durchführung:

Folgendes Szenario wird ausgewählt:

Patientanzahl	2
Datum und Zeit des Notrufs	23. Juli 2014 - 20:38 Uhr
Name des Ersthelfers	Maria Sauer
Telefonnumer des Ersthelfers	+49525170256

1. Patient:

Patient ansprechbar	Ja
Notfallsituation	Kindernotfall
Erkrankungen	Krampfanfall
Verletzungen	Keine

2. Patient:

Patient ansprechbar	Nein
Atmung vorhanden	Nein

Ergebnis:

Die Abbildungen 65 und 66 zeigen, dass die Notrufinformationen erfolgreich übertragen werden.

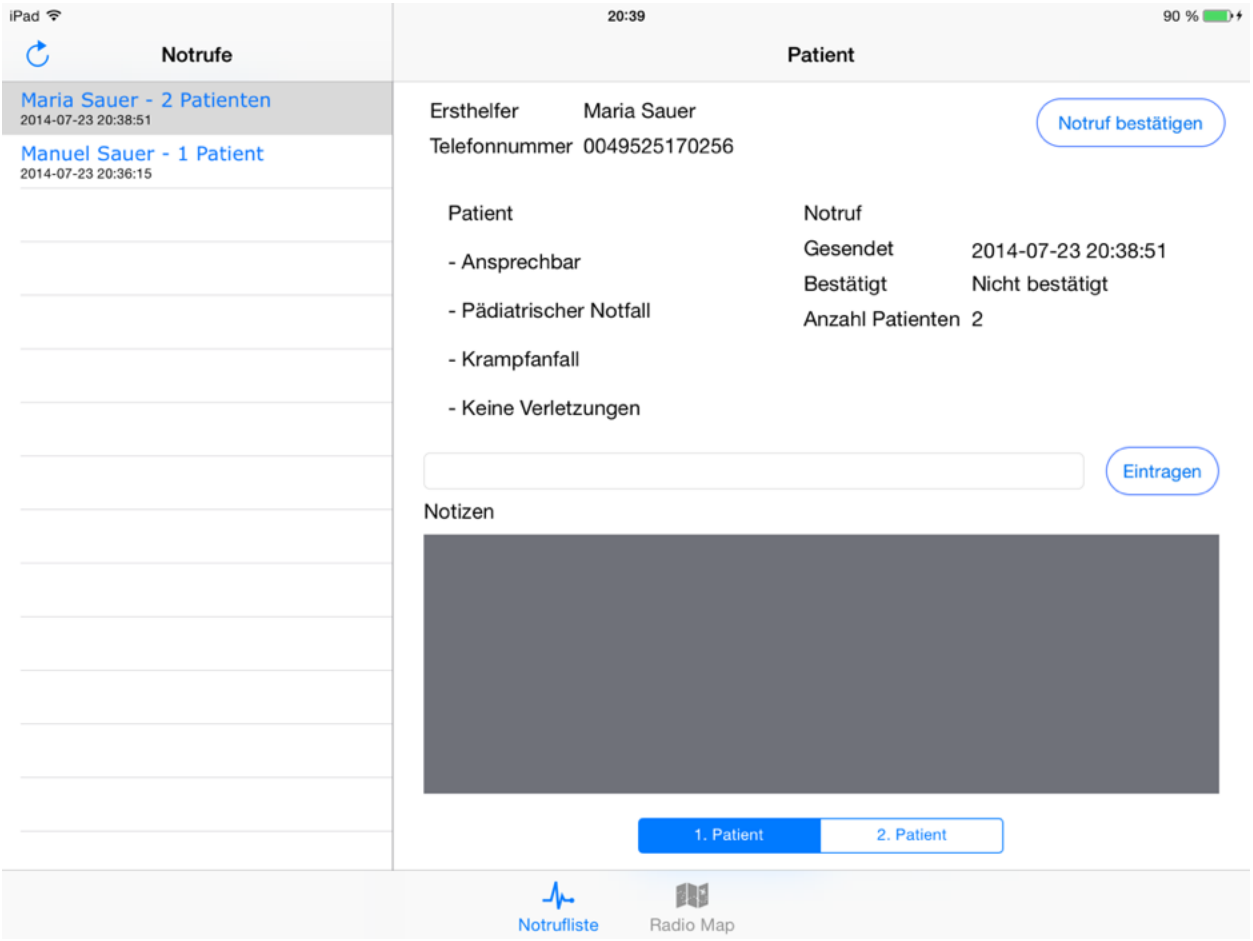


Abbildung 65: Übermittlung der Notrufinformationen

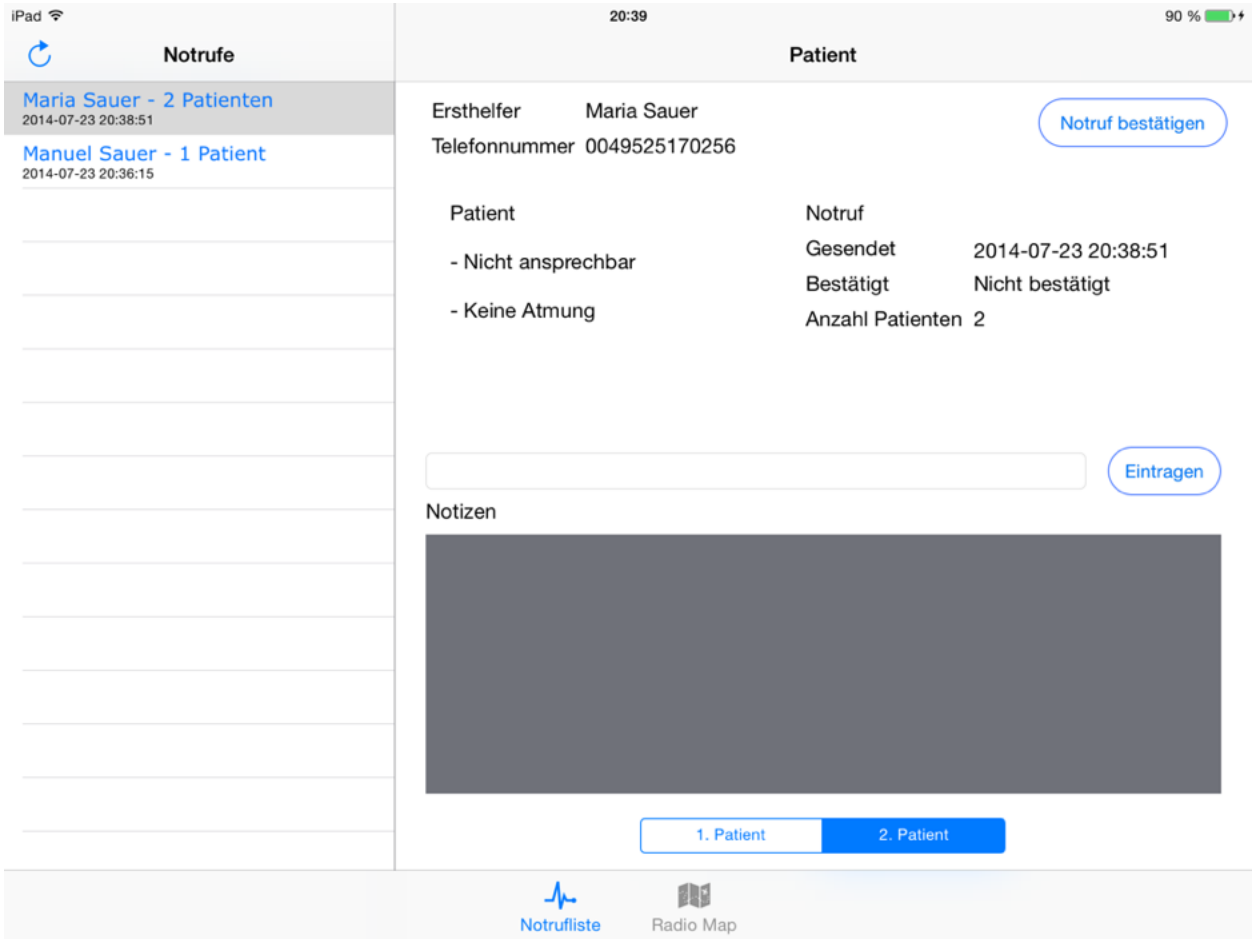


Abbildung 66: Übermittlung der Notrufinformationen

Test mit 4 Patienten:

Durchführung:

Folgendes Szenario wird ausgewählt:

Patientanzahl	4
Datum und Zeit des Notrufs	23. Juli 2014 - 20:41 Uhr
Name des Ersthelfers	Lisa Mull
Telefonnumer des Ersthelfers	05251643806

Ergebnis:

Die Abbildung 67 zeigt, dass der Notruf erfolgreich übermittelt wurde.

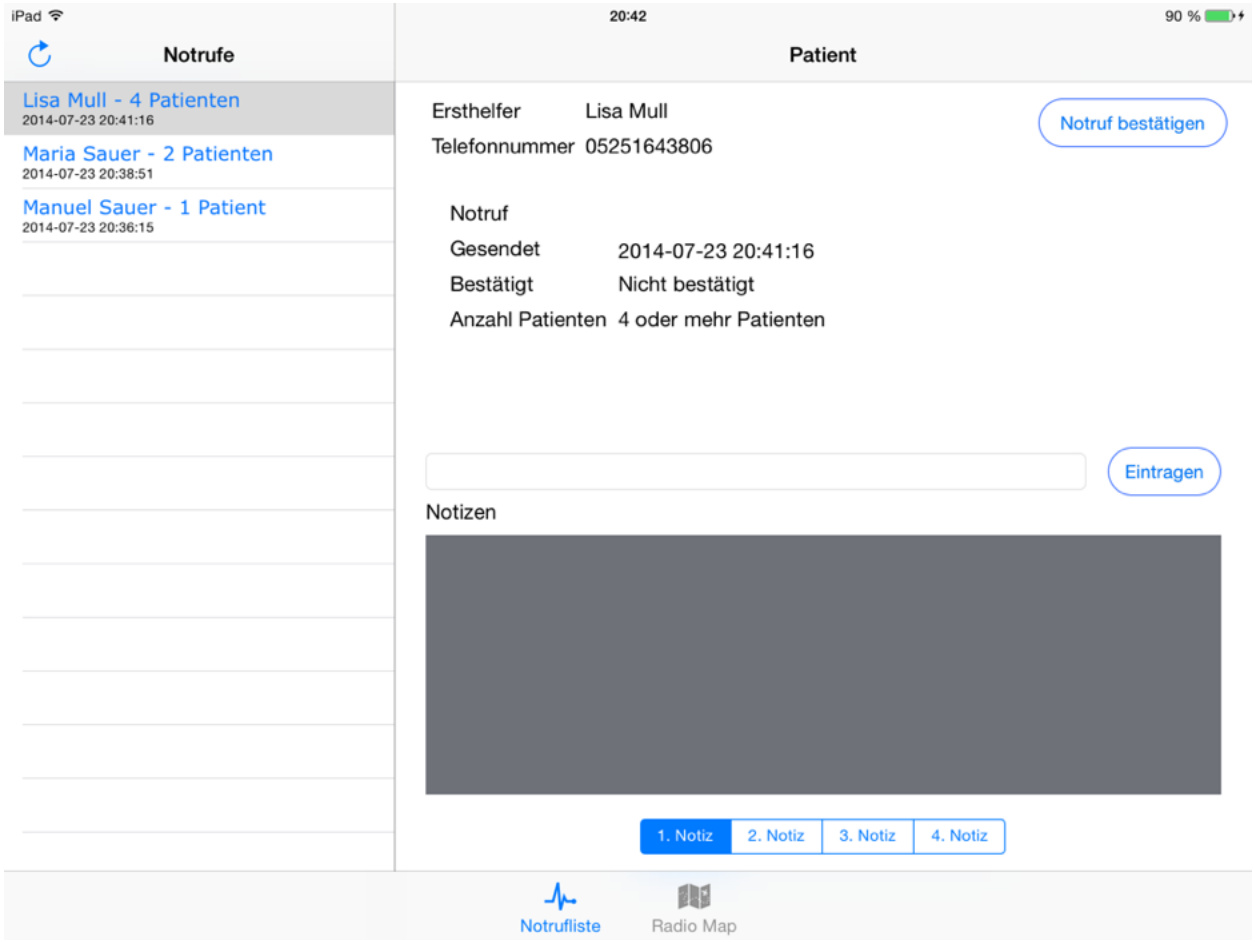


Abbildung 67: Übermittlung der Notrufinformationen

7.2 Benachrichtigung über einen neuen Notruf

Beschreibung:

Diese Tests sollen zeigen, dass die Benachrichtigung über einen neuen Notruf sowohl im inaktiven Zustand des iPads, wie auch bei einem in Benutzung befindlichen iPad (aktivem iPad) mit geschlossener und geöffneter Sanitäter-App funktioniert. Des Weiteren soll überprüft werden, ob die Anzahl der nicht bestätigten Notrufe, beim Eintreffen eines neuen Notrufs, oben am rechten Rand des Symbols der Sanitäter-App richtig angezeigt wird.

Durchführung:

Zur Durchführung des Tests werden die Sanitäter über die Helfer-App alarmiert. Das iPad ist dabei einmal inaktiv und zweimal aktiv. Im aktiven Zustand des iPads, ist die Sanitäter-App einmal geöffnet und einmal geschlossen.

Ergebnis:

Das iPad ist gesperrt:

Das iPad wird aktiv und es ertönt ein akustisches Signal. Zudem erscheint eine Nachricht mit dem Hinweis, dass ein neuer Notruf eingetroffen ist.

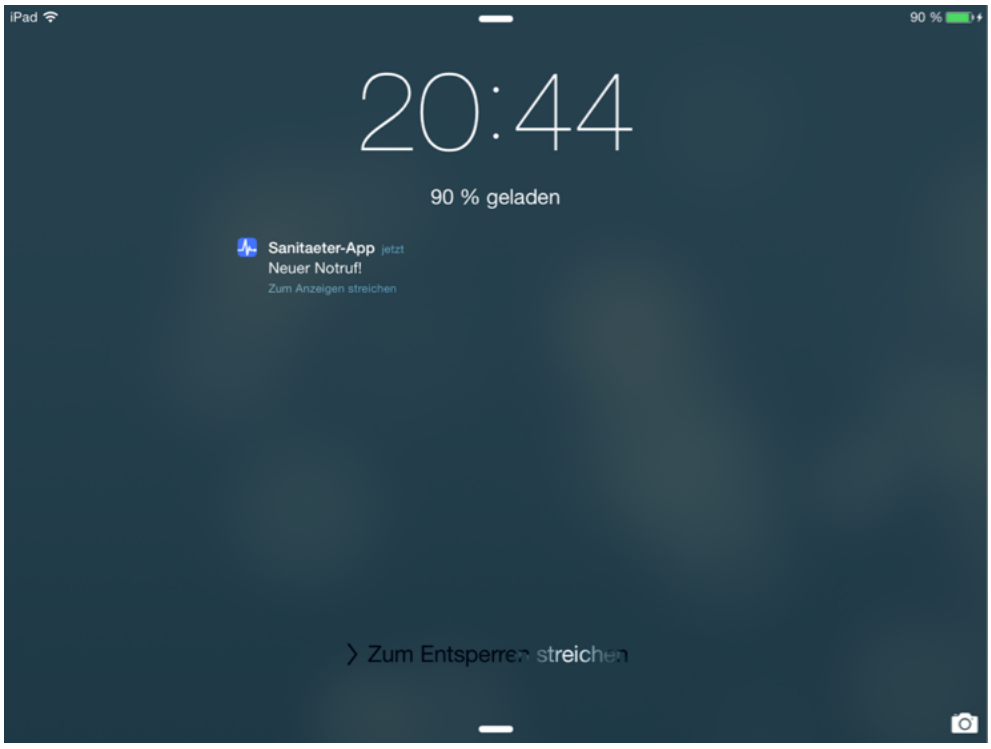


Abbildung 68: Neuer Notruf - iPad ist gesperrt

Das iPad ist aktiv. Die Sanitäter-App ist geschlossen:

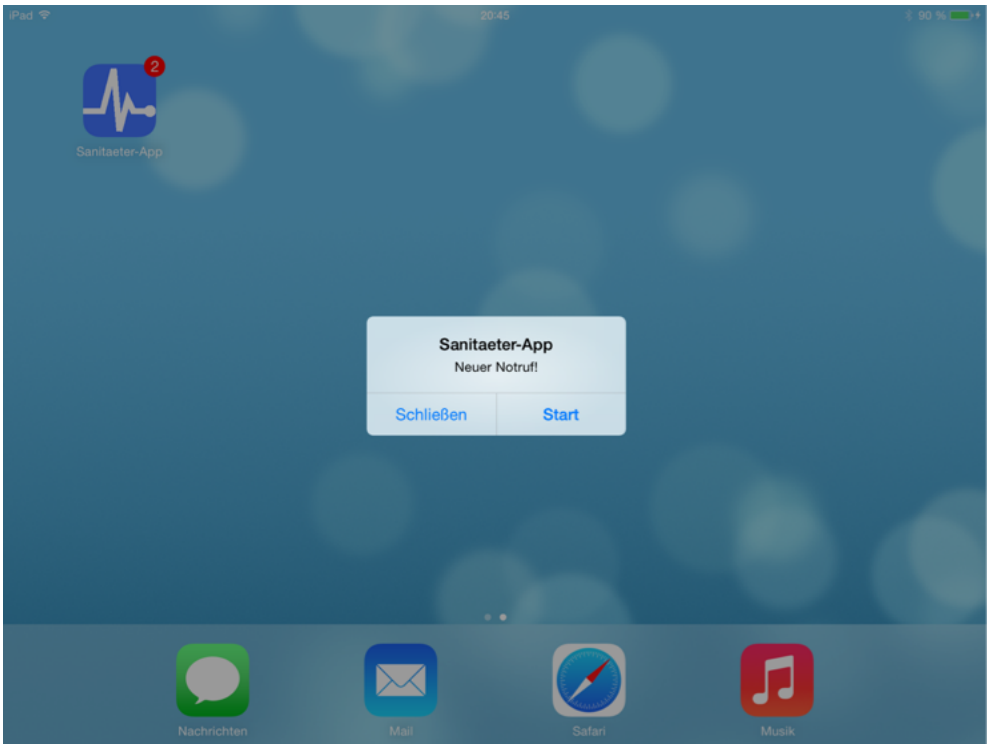


Abbildung 69: Neuer Notruf - Sanitäter-App ist geschlossen

Es erscheint ein Alert, in dem die Sanitäter über einen neuen Notruf informiert werden. Zusätzlich ertönt ein akustisches Signal. In der rechten oberen Ecke des Buttons der Sanitäter-App erscheint eine rote 2, da aktuell zwei Notrufe nicht bestätigt wurden. Dies zeigt die Abbildung 70, in der die nicht bestätigten Notrufe blau unterlegt sind.

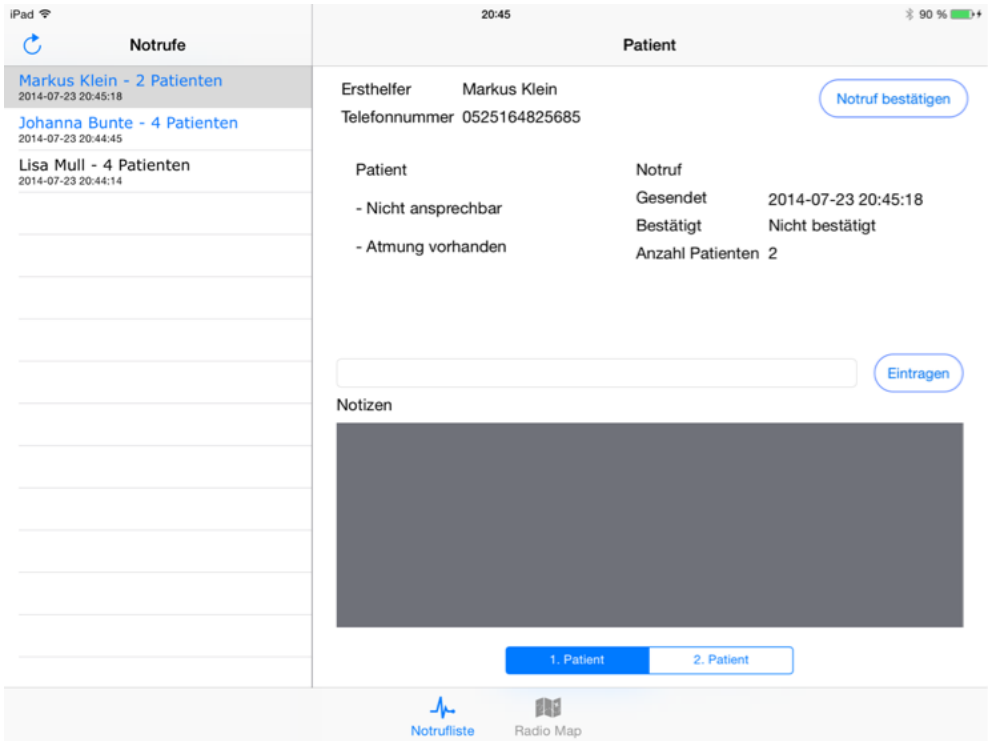


Abbildung 70: Nicht bestätigte Notrufe

Die Sanitäter-App ist geöffnet:

Es erscheint ein Alert, welcher die Sanitäter über einen neu eingetroffenen Notruf informiert (Abbildung 71). Durch Klicken des Aktualisieren Buttons wird der neue Notruf heruntergeladen.

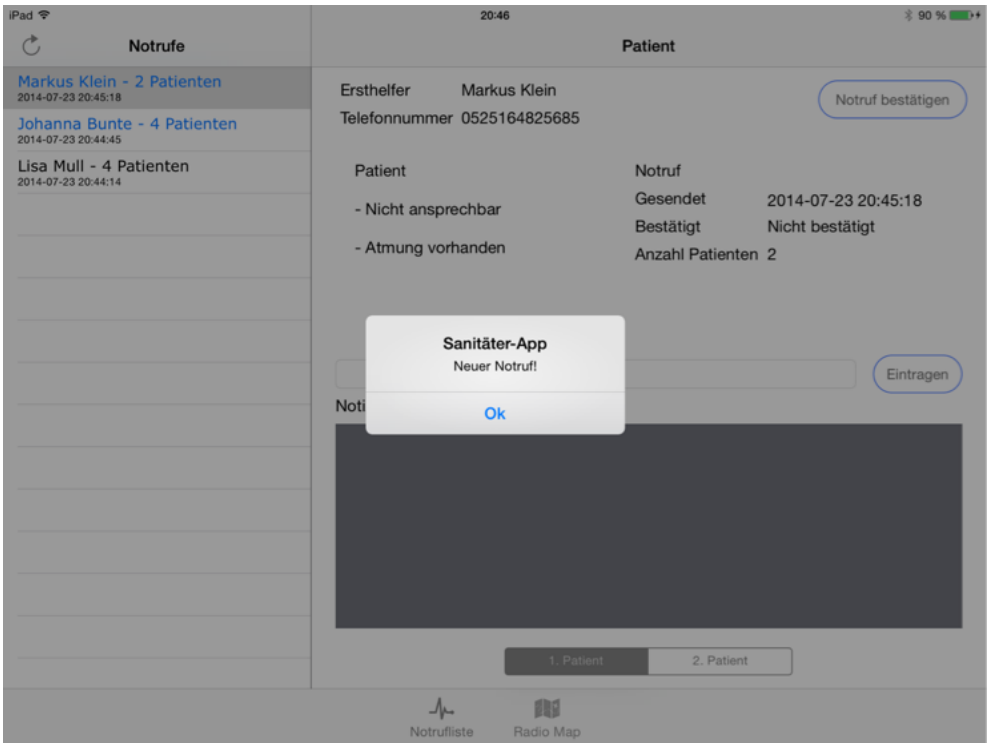


Abbildung 71: Neuer Notruf - Sanitäter-App ist geöffnet

7.3 Bestätigung eines Notrufs

Beschreibung:

In diesem Test wird untersucht, ob der Notruf erfolgreich bestätigt wird und der Ersthelfer eine Bestätigungsnachricht erhält.

Durchführung:

Dazu wird ein zuvor gesendeter Notruf in der Sanitäter-App bestätigt.

Ergebnis:

Die Abbildung 72 zeigt, dass der Notruf in der Sanitäter-App erfolgreich bestätigt wird. Die Sanitäter erhalten eine Nachricht, die Bestätigungszeit wird angezeigt. Des Weiteren werden der Name des Ersthelfers und die Patientenanzahl in der Notrufliste in der Farbe schwarz angezeigt.
Zudem zeigt die Abbildung 73, dass der Ersthelfer ebenfalls eine Nachricht über den betätigten Notruf erhält. Des Weiteren wird eine rote 1 in der oberen, rechten Ecke des Symbols der Helfer-App angezeigt (Abbildung 74).

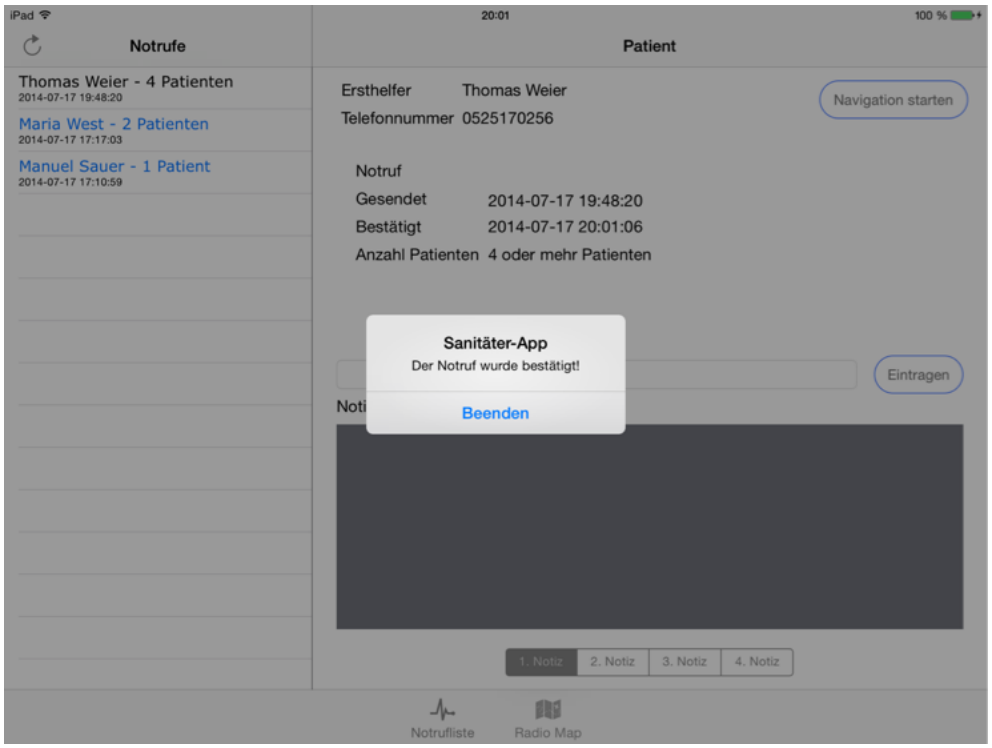


Abbildung 72: Bestätigung des Notrufs

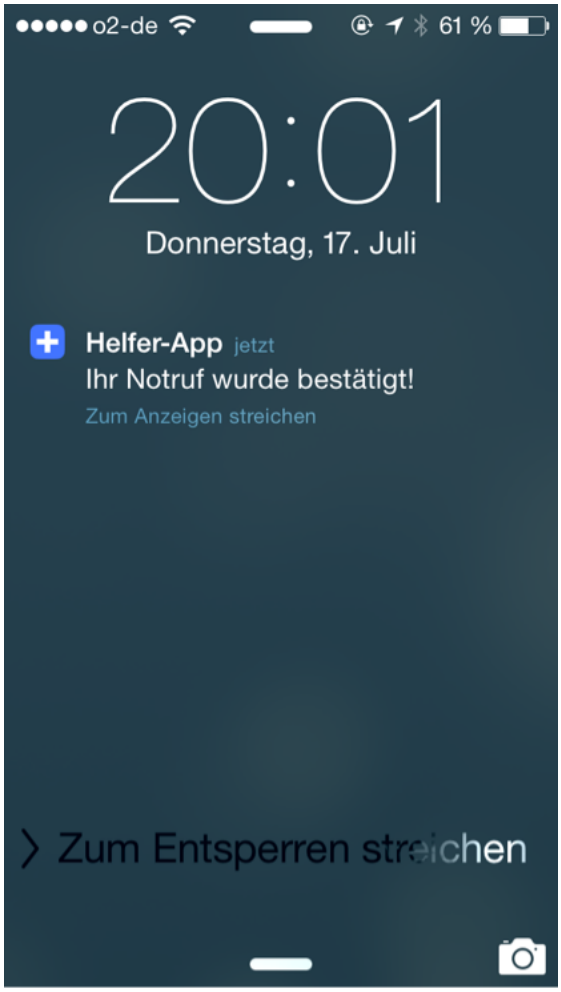


Abbildung 73: Bestätigung des Notrufs

Abbildung 74: Bestätigung des Notrufs

7.4 Ermittlung des Notfallorts

Beschreibung:

In diesem Test wird untersucht, ob der Standort des Ersthelfers erfolgreich ermittelt und an die Sanitäter-App übertragen wird. Des Weiteren wird die Genauigkeit der Standortermittlung untersucht.

Durchführung:

Für diesen Test wird der Standort von verschiedenen Orten innerhalb der Radio Map mit Hilfe der Helfer-App ermittelt und an die Sanitäter-App übertragen.

Ergebnis:

Standort	Berechneter Standort	Übertragender Standort	Euklidische Distanz: Standort - Berechneter Standort
(0 I 12)	(0 I 16)	(0 I 16)	4
(2 I 2)	(2 I 2)	(2 I 2)	0
(4 I 8)	(0 I 10)	(0 I 10)	4,47
(4 I 16)	(6 I 16)	(6 I 16)	2
(6 I 2)	(2 I 2)	(2 I 2)	4
(6 I 10)	(2 I 2)	(2 I 2)	8,94

Es fällt auf, dass von den sechs gemessenen Positionen nur einmal die korrekte Position ermittelt wird. Dies ist auf die Schwankungen der Estimote Beacons zurückzuführen. Allerdings sind fünf der bestimmten Position nur 4,50 m von der eigentlichen Position entfernt. Bei einer Großveranstaltung ist diese Genauigkeit zur Lokalisierung eines Notfallorts ausreichend.
Auffällig ist ebenfalls, dass dreimal der Standort (2I2) ermittelt wird, obwohl die Positionsbestimmung nur einmal vom Punkt (2I2) durchgeführt wird. Dies könnte auf einen Ausreißer bei der Erstellung der Radio Map im Punkt (2I2) hindeuten. Dies sollte von den Sanitätern beobachtet werden, damit die Punkte mit starken Schwankungen erneut gemessen werden können.

7.5 Ausrichtung des Navigationspfeils

Beschreibung:

In diesem Test wird untersucht, ob der Navigationspfeil korrekt ausgerichtet wird.

Durchführung:

Die Ausrichtung des Pfeils erfolgt in der Helfer- und Sanitäter-App identisch. Aus diesem Grund wird nur die Helfer-App getestet.
Die Positionsbestimmung mittels iBeacon ist in diesem Test nicht von Bedeutung, deswegen werden fiktive Startpunkte und Notausgänge als Zielpunkte vorgegeben. Die jeweiligen Start- und Zielpunkte sind auf den Abbildungen bei den Ergebnissen zu sehen. Das iPhone wird nicht Richtung Norden gehalten, deswegen ist eine Justierung nach Norden von 248° erforderlich.

Es werden alle möglichen Szenarien zur Pfeiljustierung untersucht:

1) Start.x > Ziel.x und Start.y = Ziel.y
2) Start.x > Ziel.x und Start.y > Ziel.y
3) Start.x > Ziel.x und Start.y < Ziel.y
4) Start.x < Ziel.x und Start.y = Ziel.y
5)Start.x < Ziel.x und Start.y > Ziel.y
6) Start.x < Ziel.x und Start.y < Ziel.y
7) Start.x = Ziel.x und Start.y = Ziel.y
8) Start.x = Ziel.x und Start.y > Ziel.y
9) Start.x = Ziel.x und Start.y < Ziel.y

Ergebnis:

Folgende Abbildungen zeigen die Ergebnisse der Tests:

Abbildung 75: Ergebnis Szenario 1

Abbildung 76: Ergebnis Szenario 2

Abbildung 77: Ergebnis Szenario 3

Abbildung 78: Ergebnis Szenario 4

Abbildung 79: Ergebnis Szenario 5

Abbildung 80: Ergebnis Szenario 6

Abbildung 81: Ergebnis Szenario 7

Abbildung 82: Ergebnis Szenario 8

Abbildung 83: Ergebnis Szenario 9

Die Abbildungen zeigen, dass die Navigationspfeile in die richtige Richtung weist. Allerdings zeigt z.B. der Pfeil in der Abbildung 83 nicht genau nach vorne, obwohl sich das Ziel genau vor dem Tester befindet. Dies hängt mit der Ausrichtung nach Norden zusammen. In einer weiteren Version der Helfer- und Sanitäter-App könnten diese minimalen Gradunterschiede herausgefiltert werden.

7.6 Distanz zwischen Start- und Zielpunkt

Beschreibung:

In diesem Test wird überprüft, ob die Distanz zwischen dem Start- und Zielpunkt korrekt berechnet wird.

Durchführung:

Zuerst wird die Distanz zwischen der Position der Sanitäter und dem Notfallort in der Sanitäter-App überprüft, anschließend die Distanz zwischen der Position des Ersthelfers und dem nächsten Notausgang in der Helfer-App.

Ergebnis:

Sanitäter-App:

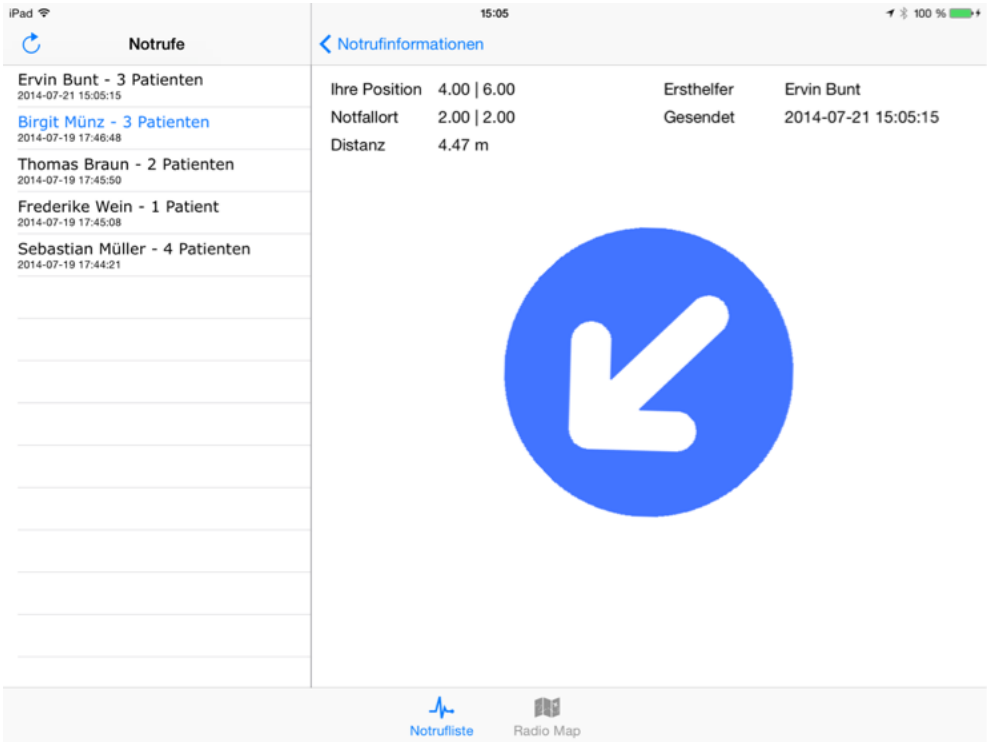


Abbildung 84: Distanzermittlung in der Sanitäter-App

Die Abbildung 84 zeigt als Position der Sanitäter (4 l 6) und als Position des Notfallorts (2 l 2) an. Die euklidische Distanz zwischen den beiden Punkten beträgt:

$$\sqrt{(p1.x\text{-}p2.x)^2 + (p1.y\text{-}p2.y)^2} = \sqrt{(4\text{-}2)^2 + (6\text{-}2)^2} = 4{,}47$$

Die Distanz wird korrekt ermittelt.

Helfer-App:

Die Abbildung 85 zeigt für die Position des Ersthelfers (4 I 10). Für die Position des nächsten Notausgangs wird (6,40 I 10,80) ermittelt.

Abbildung 85: Distanzermittlung in der Helfer-App

Die Distanz zwischen den beiden Punkten beträgt:

$$\sqrt{(p1.x\text{-}p2.x)^2 + (p1.y\text{-}p2.y)^2} = \sqrt{(4\text{-}6{,}40)^2 + (10\text{-}10{,}80)^2} = 2{,}5298$$

Die Distanz wird korrekt ermittelt.

7.7 Navigation zum Notfallort

Beschreibung:

In diesem Test wird die Genauigkeit der Navigation mittels der Sanitäter-App zu mehreren Notfallorten untersucht. Dazu werden zwei Testreihen durchgeführt. In der ersten Testreihe ist der Notfallort vorgegeben. In der zweiten Testreihe wird der Notfallort durch die Helfer-App bestimmt.

Erste Testreihe:

Durchführung:

In diesem Test lädt die Sanitäter-App Notfallinformationen von dem Webserver, führt eine Positionsbestimmung durch, und richtet den Navigationspfeil aus. Lediglich der Notfallort wird nicht durch die Helfer-App bestimmt, sondern vorher festgelegt. Der Notfallort befindet sich bei der Position N(4 l 10). Als Startposition werden die Punkte S1(2 l 2) und S3(0 l 14) gewählt.

Ergebnis:

Die Bilder zeigen zuerst eine Ausrichtung des iPads nach Norden. Anschließend mit Blick Richtung N(4 l 10).

S1(2 | 2):

Die Abbildung 86 zeigt, dass die Position korrekt ermittelt wurde und der Navigationspfeil in die richtige Richtung zeigt. Bei Drehung des iPads in Richtung N(4 | 10) dreht sich der Navigationspfeil ebenfalls (Abbildung 87). Er zeigt erneut in die richtige Richtung.

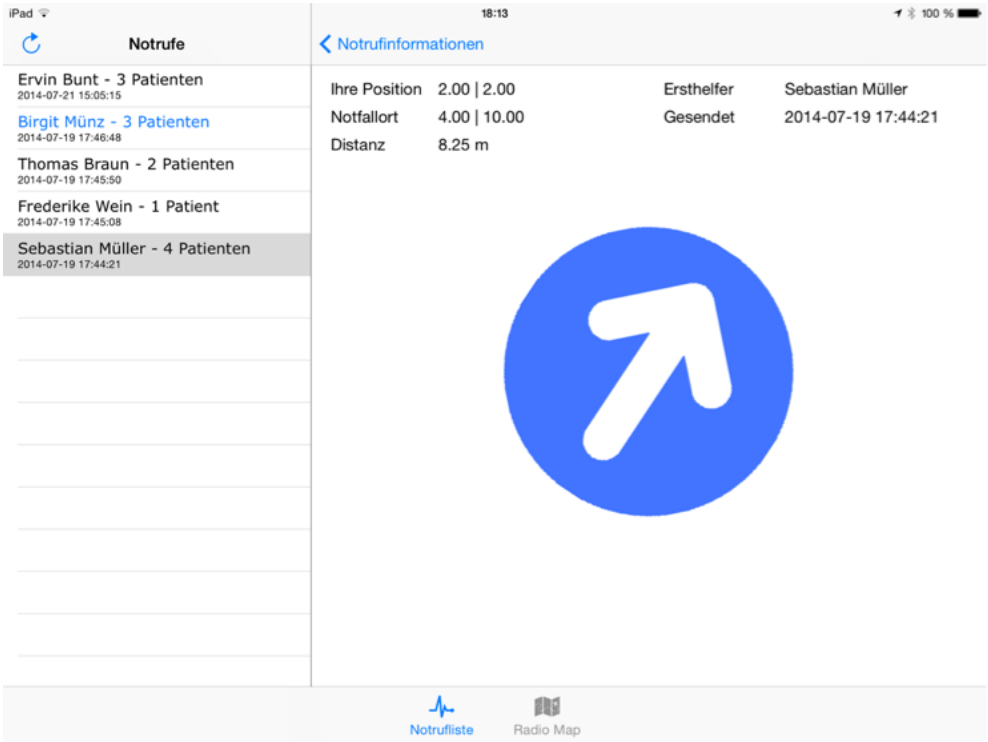


Abbildung 86: Navigation zum Notfallort

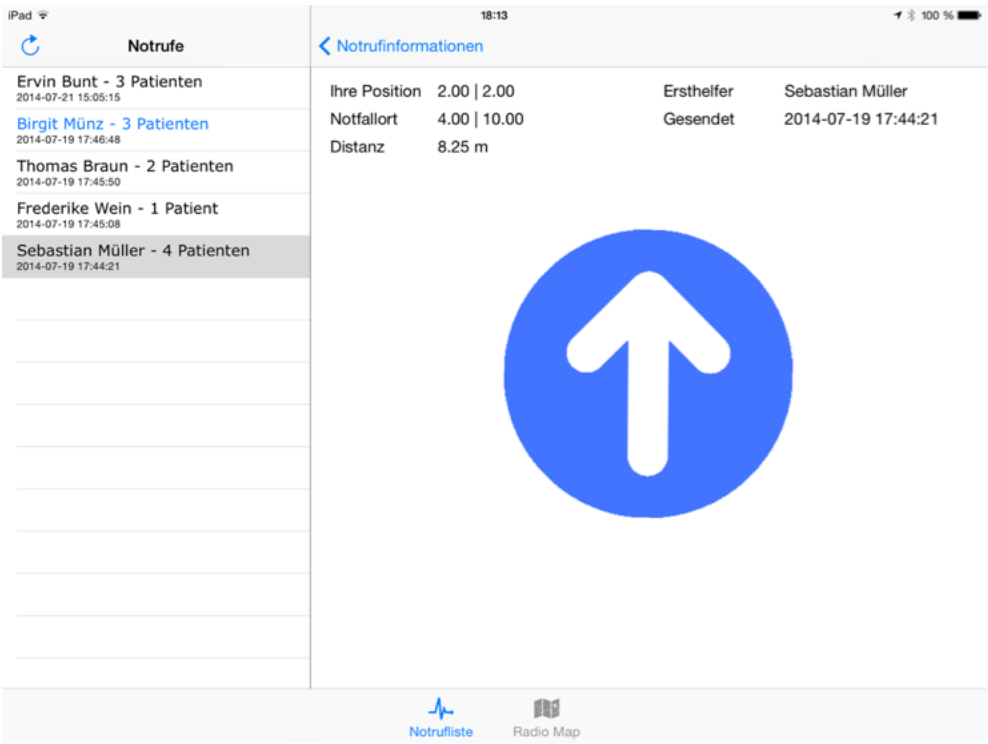


Abbildung 87: Navigation zum Notfallort

S3(0 | 14):

Die Abbildung 88 zeigt das iPad ausgerichtet Richtung Norden auf der Position S3(0 | 14). Der Standort wurde leicht falsch ermittelt. Die falsche Berechnung reicht nicht aus, um den Navigationspfeil in eine falsche Richtung weisen zu lassen. Auch bei der Drehung des iPads Richtung N(4 | 10) weist der Navigationspfeil weiterhin in die richtige Richtung (Abbildung 89).

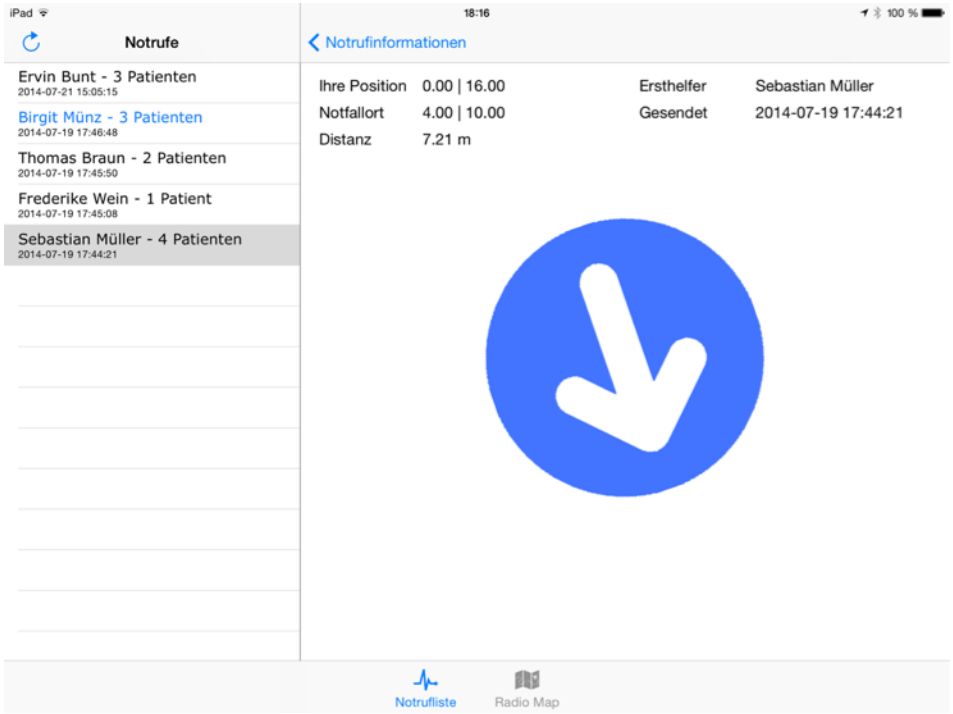


Abbildung 88: Navigation zum Notfallort

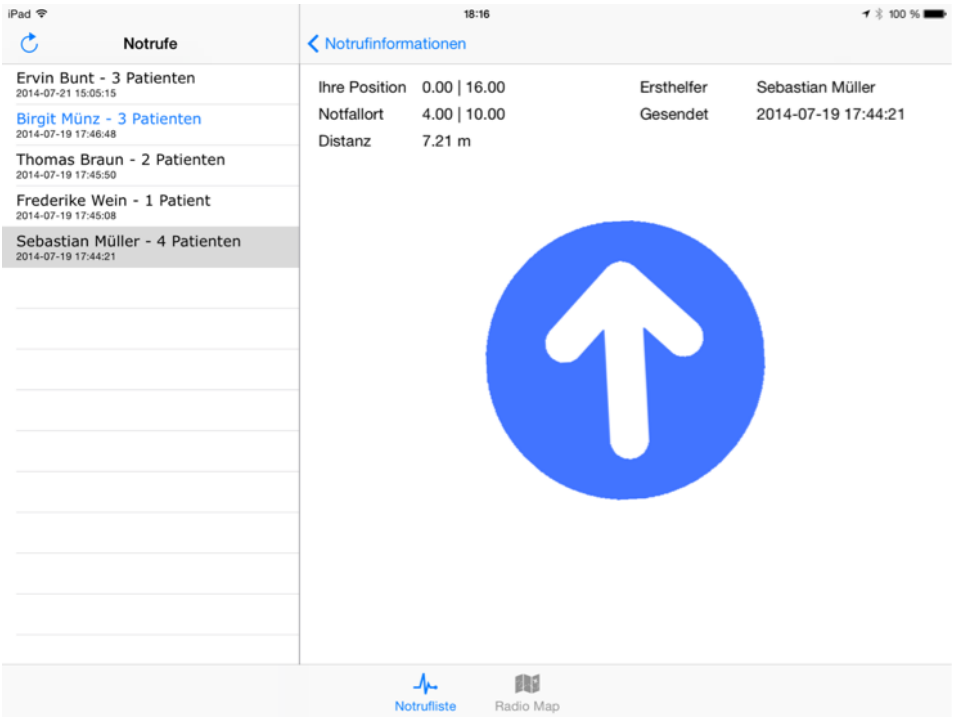


Abbildung 89: Navigation zum Notfallort

Zweite Testreihe:

Diese Testreihe ist zur ersten Testreihe identisch, bis auf, das für diese Testreihe der Notfallort nicht vorgegeben ist, sondern durch die Helfer-App ermittelt wird. Ermittelt wird der Notfallort von den Positionen N1(2 | 2), N2(6 | 16) und N3(0 | 14). Die Sanitäter befinden sich dabei für den Notfallort N1 auf der Position S1(4 | 8), für N2 auf der Position S2(0 | 0) und für N3 auf der Position S3(4 | 2).

Ergebnis:

N1(2 | 2) - S1(4 | 8):

Für die Position N1 hat die Helfer-App den Standort (0 | 0) ermittelt. Die Sanitäter-App ermittelt für S1 den korrekten Standort (4 | 8). Die Abbildung 90 zeigt, dass der Navigationspfeil bei einer Ausrichtung des iPads in Richtung Norden in die korrekte Richtung weist.

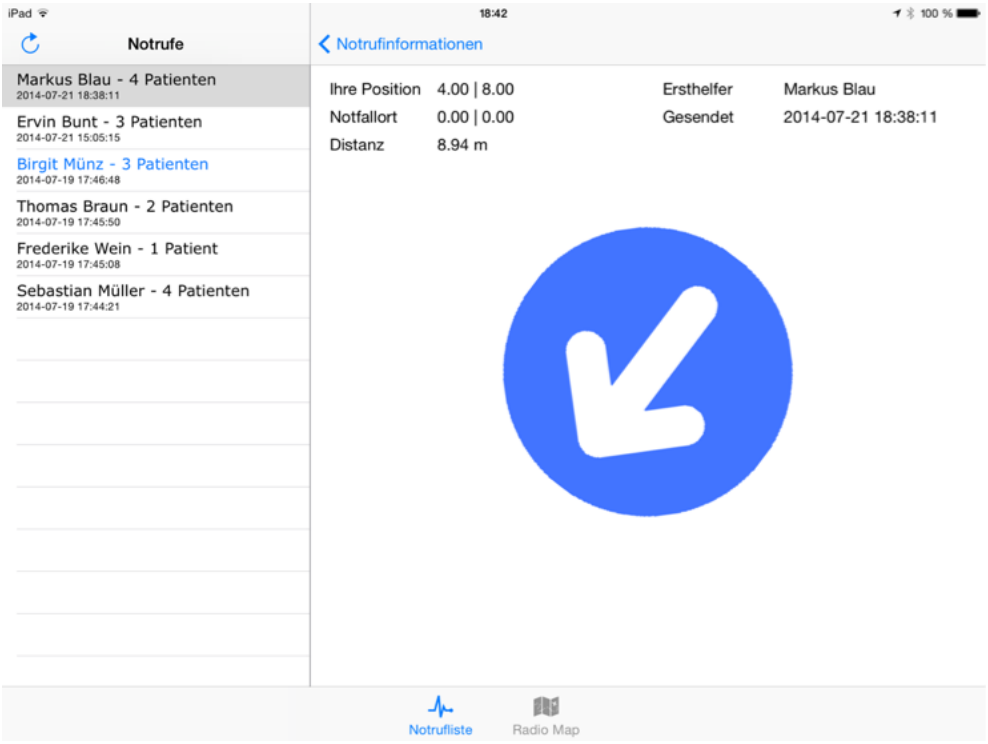


Abbildung 90: Navigation zum Notfallort -zweite Testreihe

N2(6 | 16) - S2(0 | 0):

Für den Notfallort N2 wird die falsche Position (0 | 16) ermittelt. Der Standort der Sanitäter wird korrekt mit (0 | 0) bemessen. Die Abbildung 91 zeigt, dass der Navigationspfeil fast in die richtige Richtung zeigt. Die korrekte Richtung wäre nicht nach Norden, sondern Nordosten.

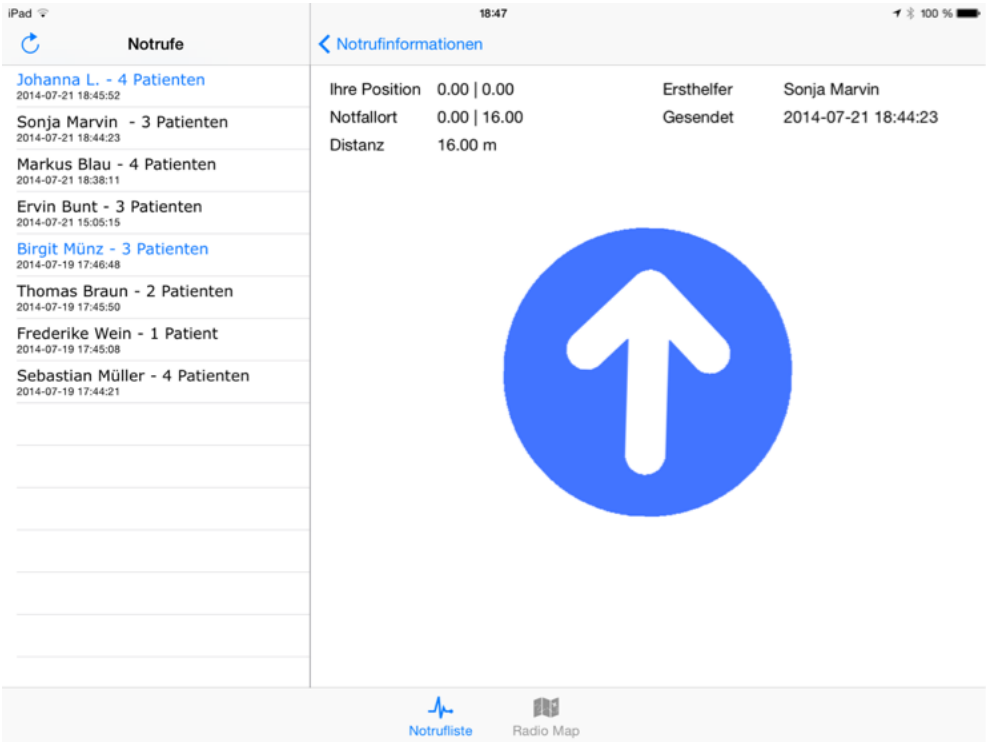


Abbildung 91: Navigation zum Notfallort - zweite Testreihe

N3(0 | 14) - S3(4 | 2):

Die Abbildung 92 zeigt, dass für N3 (0 | 14) die Position (0 | 16) ermittelt wird. Die ermittelte Position weicht nur minimal von N3 ab. Für S3 (4 | 2) wird die Position (0 | 0) ermittelt. Trotz der fehlerhaften Positionsermittlungen weist der Navigationspfeil fast in die korrekte Richtung Nordwesten.

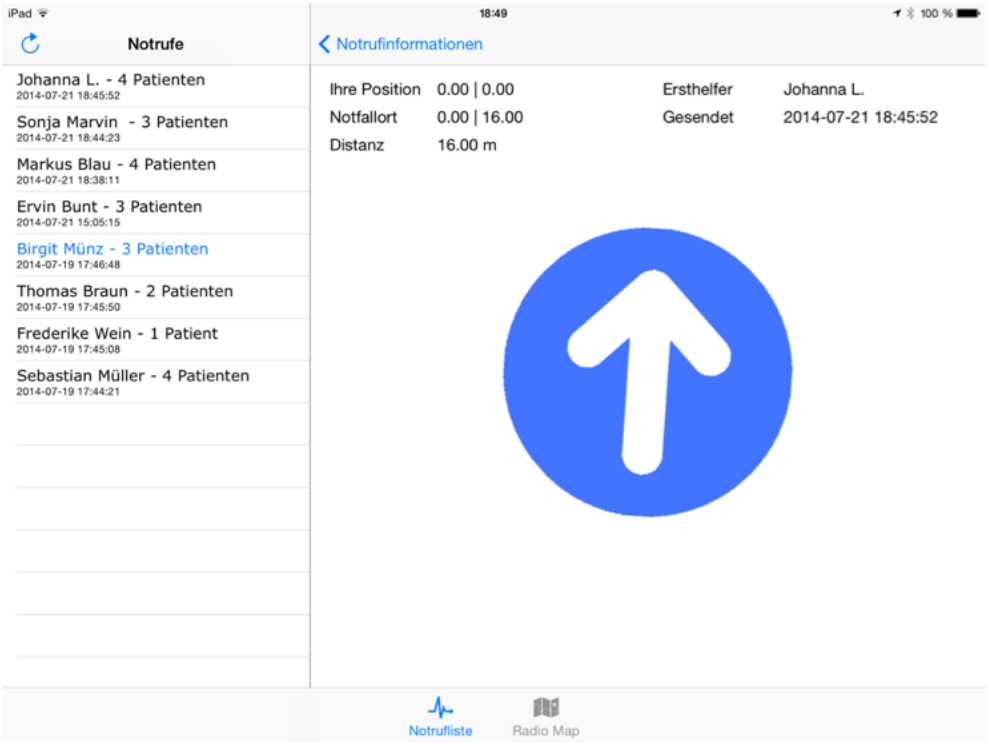


Abbildung 92: Navigation zum Notfallort - zweite Testreihe

7.8 Ermittlung des Notausganges

Beschreibung:

In diesem Test wird untersucht, ob die Ermittlung des nächsten Notausganges fehlerfrei funktioniert.

Durchführung:

Dazu wird von verschiedenen Positionen im Raum der nächste Notausgang ermittelt. Insgesamt sind drei Notausgänge mit den Positionen N1(0 | 4), N2(4,80 | 0) und N3(6,40 | 10,80) vorhanden.

Ergebnis:

Es werden nur drei von sechs Notausgängen korrekt ermittelt. Dies hängt jedoch nicht mit der Berechnung der nächsten Notausgänge zusammen, sondern mit der falschen Ermittlung des Standorts durch die Helfer-App.

Standort	Berechneter Standort	Berechneter Notausgang	nächster Notausgang
(0 | 12)	(0 | 16)	(6,40 | 10,80)	(6,40 | 10,80)
(2 | 2)	(2 | 2)	(0 | 4)	(0 | 4)
(4 | 8)	(0 | 10)	(0 | 4)	(6,40 | 10,80)
(4 | 16)	(0 | 16)	(6,40 | 10,80)	(6,40 | 10,80)
(6 | 2)	(0 | 0)	(0 | 4)	(4,80 | 0)
(6 | 10)	(0 | 10)	(0 | 4)	(6,40 | 10,80)

7.9 Navigation zum nächsten Notausgang

Beschreibung:

In diesem Test wird die Genauigkeit der Navigation zum nächsten Notausgang untersucht. Die Helfer-App führt dazu eine Positionsbestimmung, die Ermittlung des nächsten Notausganges und die Ausrichtung des Navigationspfeils durch.

Durchführung:

Es wird eine Navigation mit der Helfer-App zu den Notausgängen N1(0 | 4), N2(4,80 | 0) und N3(6,40 | 10,80) durchgeführt. Die Startpositionen des iPhones sind dabei S1(0 | 12), S2(4 | 8) und S3(6 | 2).

Ergebnis:

Abbildung 93: Navigation zum Notausgang

Für die Position S1(0 | 12) ermittelt die Helfer-App die Position (0 | 16) (Abbildung 93). Der nächste Notausgang für S1 ist mit einer Entfernung von 6,51 m der Notausgang N3(6,40 | 10,80), dieser wird auch korrekt angezeigt. Ebenfalls fast korrekt ist die Richtung zum nächsten Notausgang, wobei der Navigationspfeil etwas mehr in die westliche Richtung, als in die südliche weisen sollte.

Die Abbildung 94 zeigt, dass für die Position S2(4 | 8) die Position (2 | 2) ermittelt wird. Dadurch wird der nächste Notausgang mit N1(0 | 4) mit einer Entfernung zu S2(4 | 8) von 5,66 m falsch angegeben. Der nächste Notausgang für die Position S2(4 | 8) befindet sich bei N3(6,40 | 10,80) mit einer Entfernung von 3,69 m. Durch den falsch ermittelten Standort zeigt der Navigationspfeil in eine falsche Richtung. Korrekt ist nicht Nordwesten sondern Nordosten.

Abbildung 94: Navigation zum Notausgang

Abbildung 95: Navigation zum Notausgang

Ähnliches gilt für den Test dessen Ergebnis die Abbildung 95 zeigt. Hier wird für S3(6 | 2) der Standort (2 | 2) und damit auch der falsche Notausgang ermittelt. Korrekt ist für die Position S3 der Notausgang N2(4,80 | 0) mit einer Entfernung von 2,33 m zu S3 und nicht der ermittelte Notausgang N1(0 | 4) mit einer Entfernung von 6,32 m zu S3(6 | 2). Trotz der falsch ermittelten Position und dem damit falsch berechneten Notausgang zeigt der Navigationspfeil in die richtige Richtung. Dies hängt mit der Position von N2 zusammen. Würde sich der Notausgang an einer anderen Position z.B. (7 | 0) befinden, würde der Navigationspfeil in die falsche Richtung weisen. Das Anzeigen der richtigen Richtung ist in diesem Test demnach Zufall.

8 Ausblick

Bei den entwickelten Apps handelt es sich um Prototypen, sodass als Ausblick Funktionen optimiert oder ausgebaut werden können.
So kann z.B. im Bereich der Positionsbestimmung die verwendete Radio Map durch weitere Beacons ergänzt werden. Ebenso können weitere Radio Maps desselben Raums erstellt werden. Ein intelligentes System kann anschließend die optimale Radio Map auswählen oder mehrere Radio Maps kombinieren. Des Weiteren können die Radio Maps durch Interpolation verfeinert werden.
Als weitere Verbesserung kann das Ähnlichkeitsmaß zwischen Radio Map und gemessener Distanz unter dem Ziel optimiert werden, robuster gegen Ausreißer zu sein. Zudem erfolgt der Austausch der Radio Map zwischen Helfer- und Sanitäter-App durch die Sanitäter. Dieser Austausch kann in einer weiteren Version der Helfer- und Sanitäter-App durch eine automatische Synchronisation vereinfacht werden.
Auch ist es denkbar weitere Technologien zur Positionsbestimmung hinzuzuziehen. So kann ebenfalls die empfangene RSSI eines WLAN-Netzwerkes über das Fingerprinting-Verfahren ausgewertet werden. Das WLAN-Netzwerk kann zudem, anstatt des Internets, als Kommunikationsweg zwischen Server, Helfer- und Sanitäter-App dienen.

9 Zusammenfassung

Ziel dieser Arbeit ist es, die aktuellen Entwicklungen im Bereich Bluetooth Low Energy am Beispiel von iBeacon zu evaluieren und ein System zu entwickeln, welches es ermöglicht, auf einer Großveranstaltung Notrufinformationen zwischen Ersthelfern und Sanitätern auszutauschen. Im besonderen Maße erleichtert das System, unter Nutzung der iBeacon-Technologie, das Auffinden eines Patienten durch seine Lokalisierung, welche wiederum eine anschließende Navigation der Sanitäter zum Notfallort ermöglicht.
Bei der Abgrenzung der iBeacon-Technologie zu anderen Entwicklungen im Bereich der Indoor-Navigation stellt sich in der Arbeit heraus, dass sich iBeacon im besonderem Maße eignet um zu ermitteln, ob sich ein Endgerät in Reichweite eines bekannten Beacons befindet. Vor allem der geringere Energieverbrauch der Beacons durch die BLE-Technologie ist dabei nennenswert.
Allerdings ist die Technologie nicht ausgereift. So sind die gemessenen Distanzen zwischen einem Endgerät von Apple und einem Estimote Beacon sehr ungenau. Eine Positionsbestimmung nach dem Verfahren der Lateration ist dadurch nicht anwendbar.
Das Fingerprinting-Verfahren hingegen ermöglicht eine Positionsbestimmung mittels iBeacon, da das Gütekriterium des Verfahrens nicht in der Genauigkeit der gemessenen Distanz liegt, sondern bei der Reproduzierbarkeit der gemessenen Entfernungen zu einer Position. Allerdings treten bei wiederholten Messungen Schwankungen auf, sodass durch das Verfahren fehlerhafte Positionen ermittelt werden können. Eine Indoor-Navigation über das Fingerprinting-Verfahren mittels iBeacon ermöglicht somit keine genaue Navigation, dennoch dient sie als Orientierungshilfe.
Besonders zu empfehlen ist eine Navigation mittels iBeacon, wenn in einem Gebäude hauptsächlich zwischen verschiedenen Räumen unterschieden werden soll und die Positionsbestimmung innerhalb eines Raums nicht im Vordergrund steht.
Ebenso ist das Alarmieren der Sanitäter durch den Ersthelfer mittels einer App auf einer Großveranstaltung zu empfehlen. Als Nachteil ist zu sehen, dass die App im Vorfeld des Notfalls vom Ersthelfer zu installieren ist. Der Vorteil der Lokalisierung und die Möglichkeit, Informationen unabhängig von der Umgebungslautstärke austauschen zu können überwiegen jedoch.

10 Bibliografie

1. A. El-Rabbany, Introduction to GPS: The Global Positioning System, 2002, Artech House

2. Hans Dodel und Dieter Häupler, Satellitennavigation (German Edition), 2010, Springer

3. CLBeacon Class Reference, Apple Inc., 18. September 2013, https://developer.apple.com/library/IOs/documentation/CoreLocation/Reference/CLBeacon_class/Reference/Reference.html#//apple_ref/doc/c_ref/CLProximity

4. History of the Bluetooth Special Interest Group, Bluetooth SIG Inc., 2013, http://www.bluetooth.com/Pages/History-of-Bluetooth.aspx, aufgerufen am 21. Januar 2014

5. Herbert Pena, Bluetooth: The Insider's Guide to Bluetooth Technology, Bluetooth Security, Bluetooth Operation, Bluetooth Applications, Bluetooth Benefits, Bluetooth Devices, 2013, Tru Divine Publishing

6. A Look at the Basics of Bluetooth Technology, Bluetooth SIG Inc., http://www.bluetooth.com/Pages/Basics.aspx, aufgerufen am 22. Januar 2014

7. Naresh C. Gupta, Inside Bluetooth Low Energy, 2013, Artech House, Boston

8. Bluetooth Core Specification 4.1, Bluetooth Sig Inc., 03 Dezember 2013, https://www.bluetooth.org/en-us/specification/adopted-specifications

9. John Donovan, Bluetooth Goes Ultra-Low-Power, Digi-Key Corporation, http://www.digikey.com/us/en/techzone/wireless/resources/articles/bluetooth-goes-ultra-low-power.html, aufgerufen am 23. Januar 2014

10. Karin Zühlke und Hans Wiedemann, Bluetooth Low Energy: Ein Standard setzt sich durch, WEKA FACHMEDIEN GmbH, 2012, http://www.energie-und-technik.de/energieeffiziente-elektronik/artikel/89392/, aufgerufen am 04. Februar 2014

11. Bluetooth Sig Inc., Bluetooth® Smart Marks | Frequently Asked Questions, Bluetooth SIG Inc., 2014, https://www.bluetooth.org/en-us/bluetooth-brand/smart-marks-faqs, aufgerufen am 05. Februar 2014

12. Blue Sense Networks, 2014, http://bluesensenetworks.com, aufgerufen am 24. April 2014

13. Apple Inc., Location and Maps Programming Guide, 2014, https://developer.apple.com/library/ios/documentation/userexperience/conceptual/LocationAwarenessPG/RegionMonitoring/RegionMonitoring.html, aufgerufen am 27. April 2014

14. Axel Küpper, Location-based services : fundamentals and applications, 2005, Wiley, Chichester

15. Azadeh Kushki, Konstantinos N. Plataniotis, und A. N. Venetsanopoulos, WLAN positioning systems : principles and applications in location-based services, 2012, Cambridge University Press, Cambridge

16. Apple Inc., MFi Programm, 2014, https://developer.apple.com/programs/mfi/, aufgerufen am 09.04 2014

17. Estimote Inc., 2014, www.estimote.com, aufgerufen am 20.04 2014

18. Estimote Inc., EstimoteSDK for iOS 7, 2014, https://github.com/Estimote/iOS-SDK, aufgerufen am 10.04 2014

19. Wojtek Borowicz, How precise are Estimote Beacons? Will they work for indoor positioning?, 23. Mai 2014, https://community.estimote.com/hc/en-us/articles/201302836-How-precise-are-Estimote-Beacons-Will-they-work-for-indoor-positioning-, aufgerufen am 3. Juli 2014

20. Paypal, PayPal Beacon, 2014, https://www.paypal.com/us/webapps/mpp/beacon, aufgerufen am 11.04 2014

21. Heise Online, PayPal lässt Kunden automatisch in Läden "einchecken". 2013. http://www.heise.de/newsticker/meldung/PayPal-laesst-Kunden-automatisch-in-Laeden-einchecken-2063546.html, aufgerufen am 10.12.2013

22. European Space Agency (Esa), Galileo fixes Europe's position in history, 12.03.2013, http://www.esa.int/Our_Activities/Navigation/Galileo_fixes_Europe_s_position_in_history, aufgerufen am 03.04. 2014

23. Dr. Anja Köhne Und Dr. Michael Wößner, Aufbau des Datensignals, Dr. Anja Köhne und Dr. Michael Wößner, 14.03.2009, http://www.kowoma.de/gps/Signalaufbau.htm, aufgerufen am 03.04 2014

24. J. Wendel, Integrierte Navigationssysteme: Sensordatenfusion, GPS und Inertiale Navigation, 2007, Oldenbourg

25. Indoor Navigation, infsoft GmbH, http://www.infsoft.de/Produkte/Indoor-Navigation, aufgerufen am 09.04 2014

26. Aga Steczkiewicz, Ideal Beacon Placement, 2014, https://community.estimote.com/hc/en-us/articles/202041266-Ideal-Beacon-Placement, aufgerufen am 19. Juni 2014

27. Ola Puchta, RSSI, Range, Zones, and Distance Accuracy, 2014, https://community.estimote.com/hc/en-us/articles/201029223-RSSI-Range-Zones-and-Distance-Accuracy, aufgerufen am 27. Juni 2014

28. Wojtek Borowicz, RSSI, Range, Zones, and Distance Accuracy, 2014, https://community.estimote.com/hc/en-us/articles/201029223-RSSI-Range-Zones-and-Distance-Accuracy, aufgerufen am 27. Juni 2014

29. Angelika König, Erste Hilfe bei Kindernotfällen, 2011, Arbeiter-Samariter-Bund Deutschland e.V., Köln, Deutschland

30. U. Großmann, Innovative mobile Technologien und Anwendungen, 2009, Lit

31. Matthijs Hollemans, Apple Push Notification Services in iOS 6 Tutorial: Part 1/2, 23. Mai 2013, http://www.raywenderlich.com/32960/apple-push-notification-services-in-ios-6-tutorial-part-1, aufgerufen am 24. Mai 2014

11 Eidesstattliche Erklärung

Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und ist auch noch nicht veröffentlicht.

Heilbronn, 28. Juli 2014